Increasing the Legal Probative Value of Cryptographically Private Malleable Signatures

Maintaining a legally sufficient integrity protection
while achieving personal data protection
by means of authorized subsequent modifications.

Henrich C. Pöhls

Dissertation
eingereicht an der Fakultät für Informatik und Mathematik
der Universität Passau zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften

Gutachter

Prof. Dr. Joachim Posegga
Prof. Dr. Dieter Gollmann
Prof. Dr. Gerrit Hornung

Eingereicht im

Februar 2018

Dedicated to my family, my wife Yvonne,
and my son Philip.

Zusammenfassung

Die Arbeit befasst sich mit der Erarbeitung von technischen Vorgaben und deren Umsetzung in kryptographisch sichere Verfahren von datenschutzfreundlichen, veränderbaren digitalen Signaturverfahren (*private malleable signature schemes* oder MSS) zur Erlangung möglichst hoher rechtlicher Evidenz.

Im Recht werden bestimmte kryptographische Algorithmen, Schlüssellängen und deren korrekte organisatorische Anwendungen zur Erzeugung elektronisch signierter Dokumente als rechtssicher eingestuft. Dies kann zu einer Beweiserleichterung mithilfe signierter Dokumente führen. So gelten nach Verordnung (EU) Nr. 910/2014 (eIDAS) qualifiziert signierte elektronische Dokumente entweder als Anscheinsbeweis der Echtheit oder ihnen wird gar eine gesetzliche Vermutung der Echtheit zuteil. Gesetzlich anerkannte technische Verfahren, die einen solch erhöhten Beweiswert erreichen, erfüllen mithilfe von Kryptographie im wesentlichen zwei Eigenschaften: Integritätsschutz (*integrity*), also die Erkennung der Abwesenheit von unerwünschten Änderungen und die Zurechenbarkeit des unveränderten Dokumentes zum Signaturersteller (*accountability*).

Hingegen ist der größte Vorteil veränderbarer digitaler Signaturverfahren (MSS) die *„privacy“* genannte Eigenschaft: Eine autorisierte Änderung verbirgt den vorherigen Inhalt. Des Weiteren bleibt die Signatur solange valide wie ausschliesslich autorisierte Änderungen vorgenommen werden. Wird diese Eigenschaft kryptographisch nachweislich sicher erfüllt, so spricht man von einem *private malleable signature scheme*. In der Arbeit werden zwei verbreitete Formen, die sogenannten *redactable signature schemes* (RSS) und die *sanitizable signature schemes* (SSS), eingehend betrachtet. Diese erlauben vielfältige Einsatzmöglichkeiten, zum Beispiel eine autorisierte spätere Veränderung zur Wahrung von Geschäftsgeheimnissen oder zum Datenschutz: Der Unterzeichner delegiert so beispielsweise über ein *private redactable signature scheme* nur das nachträgliche Schwärzen (*redaction*). Dies schränkt die Veränderbarkeit auf das Entfernen von Informationen ein, erlaubt aber wirksam die Wahrung des Datenschutzes oder den Schutz von (Geschäfts)geheimnissen indem diese Informationen irreversibel für Angreifer entfernt werden. Die kryptographische *privacy* Eigenschaft besagt, dass es nun nicht mehr effizient möglich ist, aus dem geschwärzten Dokument Wissen über die geschwärzten Informationen zu erlangen, auch und gerade nicht für den Signaturprüfer.

Die Arbeit geht im Kern der Frage nach, ob ein MSS sowohl die kryptographische Eigenschaft *„privacy“* als auch gleichzeitig die Eigenschaften *„integrity“* und *„accountability“* mit ausreichend hohen Sicherheitsniveaus erfüllen kann. Das Ziel ist es, dass ein MSS gleichzeitig ein solch ausreichend hohen Grad an Sicherheit erfüllt, dass (1) die autorisierten nachträglichen Änderungen zum Schutze von Geschäftsgeheimnissen oder personenbezogenen Daten eingesetzt werden können, und dass (2) dem Dokument, welches mit dem speziellen Signaturverfahren signiert wurde, ein erhöhter Beweiswert beigemessen werden kann. In Bezug auf letzteres stellt die Arbeit sowohl die technischen Vorgaben, welche für qualifizierte elektronische Signaturen (nach Verordnung (EU) Nr. 910/2014) gelten, in Bezug auf die nachträgliche Änderbarkeit dar, als auch konkrete kryptographische Eigenschaften und Verfahren um diese Vorgaben kryptographisch beweisbar zu erreichen.

Insbesondere weisen veränderbare Signaturen (MSS) einen anderen Integritätsschutz als traditionelle digitale Signaturen auf: Eine signierte Nachricht darf nachträglich durch eine definierte dritte Partei in einer definierten Art modifiziert werden. Diese sogenannte autorisierte Änderung (*authorized modification*) kann auch ohne Kenntnis des geheimen Signaturschlüssels des

Unterzeichners durchgeführt werden. Bei der Verifikation der digitalen Signatur durch den Signaturprüfer bleibt der ursprüngliche Signierende und dessen Einwilligung zur autorisierten Änderung kryptographisch verifizierbar, auch wenn autorisierte Änderungen vorgenommen wurden.

Die Arbeit umfasst folgende Bereiche:

1. Analyse der Rechtsvorgaben zur Ermittlung der rechtlich relevanten technischen Anforderungen hinsichtlich des geforderten Integritätsschutzes (*integrity protection*) und hinsichtlich des Schutzes von personenbezogenen Daten und (Geschäfts)geheimnissen (*privacy protection*),

2. Definition eines geeigneten Integritäts-Begriffes zur Beschreibung der Schutzfunktion von existierenden *malleable signatures* und bereits rechtlich anerkannten Signaturverfahren,

3. Harmonisierung und Analyse der kryptographischen Eigenschaften existierender *malleable signature* Verfahren in Hinblick auf die rechtlichen Anforderungen,

4. Entwicklung neuer und beweisbar sicherer kryptographischer Verfahren,

5. abschließende Bewertung des rechtlichen Beweiswertes (*probative value*) und des Datenschutzniveaus anhand der technischen Umsetzung der rechtlichen Anforderungen.

Die Arbeit kommt zu dem Ergebnis, dass zunächst einmal jedwede (autorisierte wie auch unautorisierte) Änderung von einem kryptographisch sicheren *malleable signature* Verfahren (MSS) ebenfalls erkannt werden muss um Konformität mit Verordnung (EU) Nr. 910/2014 (eIDAS) zu erlangen. Eine solche Änderungserkennung durch die der Signaturprüfer, ohne Zuhilfe weiterer Parteien oder Geheimnisse, die Abwesenheit von autorisierten und unautorisierten Änderungen erkennt wurde im Rahmen der Arbeit entwickelt (*non-interactive public accountability (PUB)*). Diese neue kryptographische Eigenschaft wurde veröffentlicht und bereits von Arbeiten Anderer aufgegriffen. Des Weiteren werden neue kryptographische Eigenschaften und *redactable signature* und *sanitizable signature* Verfahren vorgestellt, welche zusätzlich zu dieser Änderungerkennung einen starken Schutz gegen die Aufdeckung des Orginals ermöglichen. Werden geeignete Eigenschaften erfüllt so wird für bestimmte Fälle ein technisches Schutzniveau erzielt, welches mit klassischen Signaturen vergleichbar ist. Damit lässt sich die Kernfrage positiv beantworten:

> *Private* MSS können ein Integritätsschutzniveau erreichen, welches dem rechtlich anerkannter digitaler Signaturen technisch entspricht, aber dennoch nachträgliche Änderungen autorisieren kann, welche einen starken Schutz gegen die Wiederherstellung des Orginals ermöglichen.

Abstract

This thesis distills technical requirements for an increased probative value and data protection compliance, and maps them onto cryptographic properties for which it constructs provably secure and especially private malleable signature schemes (MSS). MSS are specialised digital signature schemes that allow the signatory to authorize certain subsequent modifications, which will not negatively affect the signature verification result.

Legally, regulations such as European Regulation 910/2014 (eIDAS), 'follow-up' to long-standing Directive 1999/93/EC, describe the requirements in technology-neutral language. eIDAS states that, when a digital signature meets the full requirements it becomes a qualified electronic signature and then it "[...] shall have the equivalent legal effect of a handwritten signature [...]" [Art. 25 Regulation 910/2014]. The question of what legal effect this has with regards to the probative value that is assigned is actually not determined in EU Regulation 910/2014 but in European member state law. This thesis concentrates in its analysis on the — in this respect detailed — German Code of Civil Procedure (ZPO). Following the ZPO, a signature awards the signed document with at least a high probative value of prima facie evidence. For signed documents of official authority the ZPO's statutory rules even award evidence with a legal presumption of authenticity. This increased probative value is also awarded to electronic documents bearing electronic signatures when those conform to the eIDAS requirements. The requirements centre around the technical security goals of *integrity* and *accountability*. Technical mechanisms use cryptographic means to detect the absence of unauthorized modifications (integrity) and allow to authenticate the signed document's signatory (accountability).

However, the specialised malleable signature schemes' main advantage is a cryptographic property termed *privacy*: An authorized subsequent modification will protect the confidentiality of the modified original. Moreover, the MSS will retain a verifiable signature if only authorized modifications were carried out. If these properties are reached with provable security the schemes are called private malleable signature schemes. This thesis analyses two forms of MSS discussed in existing literature: *Redactable signature schemes* (RSS) which allow subsequent deletions, and *sanitizable signature schemes* (SSS) which allow subsequent edits. These two forms have many application scenarios: A signatory can delegate that a later redaction might take place while retaining the integrity and authenticity protection for the still remaining parts. The verification of a signature on a redacted or sanitized document still enables the verifying entity to corroborate the signatory's identity with the help of flanking technical and organisational mechanisms, e.g. a trusted public key infrastructure. The valid signature further corroborates the absence of unauthorized changes, because the MSS is still cryptographically protecting the signed document from undetected unauthorized changes inflicted by adversaries. Due to the confidentiality protection for the overwritten parts of the document following from cryptographic *privacy* the sanitization and redaction can be used to safeguard personal data to comply with data protection regulation or withhold trade-secrets.

The research question is: Can a malleable signature scheme be private to be compliant with EU data protection regulation **and at the same time** fulfil the integrity protection legally required in the EU to achieve a high probative value for the data signed?

Answering this requires to understand the protection requirements in respect to accountability and integrity rooted in Regulation 910/2014 and related legal texts. This thesis has analysed the previous Directive 1999/93/EC as well as German SigG and SigVO or UK and US laws. Besides that, legal texts, laws and regulations for the protection requirements of personal

data (or PII) have been analysed to distill the confidentiality requirements, e.g. the German BDSG or the EU Regulation 2016/679 (GDPR). Moreover, an answer to the research question entails understanding the relevant difference between regular digital signature schemes, like RSASSA-PSS from PKCS-v2.2 [422], which are legally accepted mechanisms for generating qualified electronic signatures and MSS for which the legal status was completely unknown before the thesis. Especially as MSS allow the authorized entity to adapt the signature, such that it is valid after the authorized modification, without the knowledge or use of the signatory's signature generation key. On verification of an MSS the verifying entity still sees a valid signature technically appointing the legal signatory as the origin of a document, which might — however — have undergone authorized modifications after the signature was applied.

The thesis documents the results achieved in several domains:

1. Analysis of legal requirements towards integrity protection for an increased probative value and towards the confidentiality protection for use as a privacy-enhancing-technique to comply with data protection regulation.
2. Definition of a suitable terminology for integrity protection to capture (a) the differences between classical and malleable signature schemes, (b) the subtleties among existing MSS, as well as (c) the legal requirements.
3. Harmonisation of existing MSS and their cryptographic properties and the analysis of their shortcomings with respect to the legal requirements.
4. Design of new cryptographic properties and their provably secure cryptographic instantiations, i.e., the thesis proposes nine new cryptographic constructions accompanied by rigorous proofs of their security with respect to the formally defined cryptographic properties.
5. Final evaluation of the increased probative value and data-protection level achievable through the **eight proposed cryptographic malleable signature schemes**.

The thesis concludes that the detection of any subsequent modification (authorized and unauthorized) is of paramount legal importance in order to meet EU Regulation 910/2014. Further, this thesis formally defined a public form of the legally requested integrity verification which allows the verifying entity to corroborate the absence of any unauthorized modifications with a valid signature verification while simultaneously detecting the presence of an authorized modification — if at least one such authorized modification has subsequently occurred. This property, called *non-interactive public accountability (PUB)*, has been formally defined in this thesis, was published and has already been adopted by the academic community. It was carefully conceived to not negatively impact a base-line level of privacy protection, as non-interactive public accountability had to destroy an existing strong privacy notion of transparency, which was identified as a hinderance to legal equivalence arguments. With RSS and SSS constructions that meet these properties, the thesis can give a positive answer to the research question:

> *Private* MSS can reach a level of integrity protection and guarantee a level of accountability comparable to that of technical mechanisms that are legally accepted to generate qualified electronic signatures giving an increased probative value to the signed document, while at the same time protect the overwritten contents' confidentiality.

Acknowledgements

First, I thank *Joachim Posegga* who allowed me to openly search for my research topic and then supported me to the fullest extent to explore my chosen topics in lengthy depths. I owe him great thanks for his everlasting encouragement and ongoing support. I always left the discussions with a good vibe of being supported by all means and encouraged to take the next — sometimes challenging — step. The advice he gave me during plenty of occasions was always supportive and helpful. I am grateful for the excellent time I had working on my thesis under his supervision and in his research group. Moreover, I was given the luxury to work on three international research projects with the full intensity and ability to look and work also behind their facade.

Second, I thank *Dieter Gollmann*, whom I first met during my studies at Royal Holloway. When it manifested that I was going to write about security terminology, he was my first choice as a second examiner. I thank him for his time.

Third, I thank *Gerrit Hornung* in particular for his willingness to work on this interdisciplinary thesis topic. As I am not a student of law, I am grateful for this opportunity and the time he invested.

In my life as a student and researcher I had the opportunity to meet and — closely and intensely — work with talented, hard-working and interesting people which were great colleagues, co-authors, project-co-workers and friends. The following list is sorted alphabetically; each of you probably knows what you did to deserve being mentioned: *Adam Kapovits, Alexandros G. Fragkiadakis, Anastasius Gavras, Antonio Kung, Arne Bilzhause, Bastian Braun, Benedikt Petschkuhn, Boris Rozenberg, Charles Basto Rodriguez, Chez Ciechanowicz, Chris Brzuska, Chris Mitchell, Christian Hanser, Christian Wittke, Christoph Frädrich, Daniel Calvo, Daniel Slamanig, David Derler, David Gundlegard, Denise Demirel, Elias Tragos, Elisabeth Reiter, Focke Höhne*[1], *Frank Kargl, Frank W. J. van Geelkerken, Fred Piper, George Moldovan, George Oikonomou, Gianmarco Baldini, Hermann de Meer, Jan Camenisch, Johannes Bauer, Johannes Rückert, Jorge Cueller, Kai Samelin, Klaus Brunnstein, Lars Westphal, Manuel Liedl, Marcin Wójcik, Margit Wiechmann, Marita Güngerich, Markus Karwe, M.-J. Hoffmann de Herrera, Mathieu des Noes, Max Mössinger, Meiko Jensen, Michael T. Beck, Nicolas Notario, Noëlle Rakotondravony, Peter Langendörfer, Ralf C. Staudemeyer, Ralph Herkenhöner, Ricardo Neisse, Robert Carolina, Santiago Suppan, Septimiu Nechifor, Siglinde Böck, Simone Fischer-Hübner, Solange Ghernaouti, Stefan Peters, Stephan Krenn, Thomas Länger, Thomas Gross, Thomas Loriünser, Vangelis Angelakis, Wolfgang Popp, Yod Samuel Martin, Zoya Dyka*. I hope I did not forget to mention anyone.

A special thanks goes to all people in the offices around me at the Universities of Hamburg, Royal Holloway and Passau which I did not name above: Especially, to my fellow PhD students. It was always a good and often cheerful work atmosphere while you still were forthright, honest and constructive with your critique. This helped me to improve.

Above all, thanks to my *family et al.* — old or new, near or distant, big or small — for their great — and sometimes for them nearly unbearable — support that allowed me to pursue and finish this work.

Thanks to you all !

[1] This first footnote had to be dedicated to you; the fun we had shines in some of our joint papers' footnotes.

Contents

1. Introduction, motivation, research question, scope, methodology and organisation 1

1.1. Introduction . . . 1

1.2. Motivation, problem statement and examples . . . 3

1.2.1. Problem statement . . . 3

1.2.2. Malleable signature schemes with added goals . . . 4

1.2.3. Examples for authorized subsequent modifications to documents . 7

1.3. Research question . . . 13

1.4. Scope . . . 14

1.4.1. Technical scope . . . 14

1.4.2. Legal scope . . . 15

1.4.3. Out of scope . . . 16

1.5. Methodology and interpretation of legal texts . . . 18

1.6. Organisation of this thesis . . . 20

2. Existing terms and definitions 21

2.1. Definition of document / message . . . 21

2.2. Definition of an electronic document . . . 22

2.3. Definition of cryptographic primitive, cryptographic scheme and cryptographic provable security . . . 22

2.3.1. Definition provable security . . . 22

2.3.2. Definition of a provable secure scheme . . . 23

2.3.3. Provable security proves security on the cryptographic level . . . 24

2.4. Notations for algorithmic descriptions . . . 25

2.5. Notations for a random bit . . . 25

2.6. Definition of an adversary's negligible success and efficient algorithm . . . 25

2.7. Background on the Random Oracle (RO) and limitations . . . 26

2.8. Definition of a cryptographic hash function primitive . . . 26

2.9. Definition of a *classic* digital signature scheme based on asymmetric cryptography (CDSS) . . . 27

2.9.1. Definition of an asymmetric signature system following ISO 9798 . 27

2.9.2. Algorithmic description of a CDSS . . . 27

2.9.3. Definition of correctness for DSS . . . 28

2.9.4. Definition of cryptographic unforgeability for DSS . . . 28

2.10. Definition of involved entities to bridge cryptographic and legal terminology: Signatory, Signer in DSS, Verifier, Certificate Issuer . . . 29

2.10.1. Workflow of a DSS and definition of a trusted third party (TTP) . . 31

2.10.2. Definition of accountability and Signer-accountability in a DSS . . . 33

2.11. Definition of a qualified public key certificate . . . 35

2.12. Definition of a natural person and a legal person . . . 37

2.13. Definition of hardware security module (HSM) . . . 37

2.14. Definition of (qualified) electronic signature . . . 38

2.15. Definition of personally identifying information (PII), personal data and data subject . . . 38

2.16. Definition of consent . . . 38

3. New definitions and harmonised terms and notation for malleable signatures 41
3.1. Detailed notion of integrity (see Chapter 6) 41
3.2. Definition of blocks ($m[i]$) and related terms 42
3.3. Definition of modification (see Sec. 5.5.2) 45
3.3.1. Definition of unauthorized modification (see Sec. 5.5.2.1) 45
3.3.2. Definition of authorized modification (see Sec. 5.5.2.3) 45
3.4. Definition of authorized modification and the notions 'malleable', 'redactable' and 'sanitizable' 46
3.4.1. Notion 'malleable' 46
3.4.2. Notion 'redaction' 48
3.4.3. Notion 'sanitization' 49
3.4.4. Separate security models of RSS and SSS (see Chapter 15) 50
3.5. Authorized modifications policy (ADM), immutable ($m[\mathsf{fix}]$) or admissible blocks ($m[\mathsf{ADM}]$), instructions for modifications (MOD), and all admissible messages ($\mathsf{span}_{\vdash}(m, \mathsf{ADM})$) 50
3.6. Entities and their secrets in MSS 53
3.6.1. Definition of roles for involved entities separated by their secrets . . 53
3.6.2. Secrets and trust relations between entities in MSS 56
3.7. Authorization of principals and actions 56
3.8. Adversary models, game-based property definitions and security assumptions 57
3.8.1. Games describing adversary's capabilities and the scheme's security property 57
3.8.2. Security Assumption 1 (Secret keys are not known to the adversary) 58
3.8.3. Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) 58
3.9. High-level definition of privacy for MSS (see Sec. 11.6.3 for SSS and Sec. 11.13.3 for RSS) 59
3.10. High-level definition of accountability for MSS (see Chapter 6, Sec. 11.6.6, or Sec. 13.1) 60
3.10.1. High-level definition of cryptographic transparency for MSS (see Chapter 6, Sec. 11.6.4, or Sec. 11.13.4) 61
3.10.2. Difference between the cryptographic properties transparency and public accountability for MSS (see Sec. 13.1) 63

I Proposed extended integrity definition & technical requirements for a high probative value 67

4. High probative value and statutory evidentiary presumption for electronic documents 69
4.1. Background on probative value and statutory evidentiary presumption from German Code of Civil Procedure (ZPO) 71
4.1.1. Statutory evidentiary presumption following German ZPO 72
4.1.2. Probative value of paper-based documents induced by documentary evidence ('Urkundenbeweis') following German § 416 ZPO 73
4.1.3. Probative value of paper-based documents induced by evidence taken by visual inspection ('Augenscheinsbeweis') following German § 371 ZPO 73
4.2. Probative value and statutory evidentiary presumption of electronic documents in German and UK law 74
4.2.1. Definition of electronic document 74
4.2.2. German § 371a ZPO and § 87a AO on the probative value and statutory evidentiary presumption of electronic documents 74
4.2.3. United Kingdom law on the probative value of electronic documents with a qualified signature 79
4.3. Analysis of legal definitions regarding integrity and authenticity from IT-related legislative acts in EU, German and US law 80
4.3.1. EU law: Notion of integrity in EU legislative acts in the area of general IT 81

4.3.2. German law: Integrity as a constitutional basic right 82
4.3.3. EU law: Notion of data integrity with respect to authorized modifications 83
4.3.4. US law: Notion of data integrity with respect to authorized modifications 84
4.4. EU signature law: Notions of integrity and authenticity 85
4.4.1. Terminology: Electronic vs. digital signature 86
4.4.2. EU Directive 1999/93/EC: Goal and legal effect 86
4.4.3. EU Regulation 910/2014: Goal and legal effects 87
4.4.4. EU Directive 1999/93/EC: Notions of integrity and accountability . 89
4.4.5. EU Directive 1999/93/EC: Technical process for signature generation 90
4.4.6. EU Regulation 910/2014: Notions of integrity and accountability . . 91
4.4.7. EU Regulation 910/2014: Technical process for signature creation and comparison to Directive 1999/93/EC 92
4.4.8. EU Regulation 910/2014: Remote signatures in the light of subsequent authorized modifications 94
4.5. German signature law: Analysis of the impact of authorized modifications on the probative value in German signature law 95
4.5.1. German signature law: Officially suggested interpretation of the required functionality of an electronic signature (BT-Drs 14-4987) . . 95
4.5.2. German signature law: The legality of the 'container signature' . . . 96
4.5.3. German signature law: The notion of 'subsequent change' in German signature legislation as introduced by Directive 1999/93/EC 96
4.5.4. German law: Impact of eIDAS to the treatment of subsequent modifications (especially VDG and eIDASDG) 97
4.5.5. German law: The notion of 'blanket statements' with respect to MSS's delegation functionality 98
4.6. Analysis results of legal definitions regarding integrity and a high probative value 101
4.6.1. Analysis Result 1: Legal texts acknowledge the importance of integrity 101
4.6.2. Analysis Result 2: Legal acts do not explicitly forbid authorized modifications 101
4.6.3. Analysis Result 3: European and German legal texts discuss the rules for probative value and statutory evidentiary presumption for digital/electronic documents; and they point to electronic signature legislation which describe the requirements for qualified/advanced electronic signatures in technology-neutral language 102
4.6.4. Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected 105
4.6.5. Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory 105
4.6.6. Analysis Result 6: Qualified electronic signatures on electronic documents when computed by qualified electronic signature creation device (QSCD) award the same probative value that handwritten signatures award to paper documents following member state rules 106
4.6.7. Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights 107

5. Analysis of existing technical integrity and authenticity definitions with respect to authorized modifications 109
5.1. Authorized modifications in existing technical definitions regarding integrity 109
5.1.1. Integrity considerations of secure computer systems by *Biba* 110
5.1.2. Data integrity by the Trusted Computer System Evaluation Criteria (TCSEC) 110
5.1.3. Data integrity by *Clark* and *Wilson* 111
5.1.4. Integrity by ISO 2700x series 111

5.1.5. Data integrity from ISO 7498-2 and ITU X.800 112
5.1.6. Data integrity in RFC 4949 113
5.1.7. Integrity of network messages by *Stallings* 114
5.1.8. Data integrity and veracity by *Gollmann* 115
5.1.9. Data integrity by *Bishop* 116
5.1.10. Integrity framework from ISO 10181-6 117
5.2. Authorized modifications in existing technical definitions regarding (data) origin authentication and non-reputation 118
5.2.1. Data origin authentication from ISO 7498-2 and ISO 13888-1 118
5.2.2. Data origin authentication from ITU X.800 119
5.2.3. Non-repudiation with proof of origin from ITU X.800 119
5.2.4. Non-repudiation with proof of origin from RFC 4949 120
5.2.5. Non-repudiation of origin by ISO 10181-4 and ISO 13888-1 121
5.2.6. Data origin authentication and non-repudiation by *Gollmann* 122
5.3. Authorized modifications in existing technical definitions of classic digital signature schemes (CDSS) 123
5.3.1. Germany: Catalogue of algorithms ('Algorithmenkatalog') by BNetzA 123
5.3.2. EU: Algorithms, key size and parameters report by ENISA 124
5.3.3. United States of America: Special publication 800-131A by NIST . . 125
5.3.4. Digital signatures by *Goldwasser*, *Micali*, and *Rivest* 125
5.3.5. Definition of digital signatures from ITU X.800 and ISO 7498-2 . . . 126
5.3.6. Definition of digital signature from NIST 127
5.3.7. Notion of transferability for digital signatures 128
5.4. Definitions of integrity in selected existing MSS literature 129
5.4.1. Integrity by *Miyazaki* 129
5.4.2. Integrity by *Johnson et al.* 129
5.4.3. Integrity by *Agrawal et al.* 129
5.5. Analysis results of existing technical integrity and authenticity definitions with respect to authorized modifications 130
5.5.1. Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law . 130
5.5.2. Analysis Result 9: Definition of authorized and unauthorized modification; and modification in general 133
5.5.3. Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification 136
5.5.4. Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect 137
5.5.5. Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document 138
5.5.6. Analysis Result 13: Digital signature schemes are a suitable integrity and authenticity protection mechanism 140

6. Proposed extended integrity definition 141
6.1. Integrity extended by three aspects: Allowed authorized modifications, detectability of modifications and accountability for modifications 142
6.2. 1^{st} aspect: authorized or no subsequent changes allowed (UCA, ACA, NCA) 143
6.2.1. Unauthorized subsequent changes allowed (UCA) 143
6.2.2. No subsequent changes allowed (NCA) 143
6.2.3. Authorized subsequent changes allowed (ACA) 144
6.3. 2^{nd} aspect: Detection of subsequent changes 144
6.3.1. Occurred change detection (*CD) 145
6.3.2. Potential future change detection (*FD) 146
6.3.3. Levels from *CD and *FD can be combined in one scheme 147

6.4. 3^{rd} aspect: Determine 3^{rd}-party verifiability and accountability for subsequent modifications of messages . . . 148
6.4.1. Interactive non-public (correctly-executed and non-repudiable) accountability (INT) for integrity-protected messages . . . 150
6.4.2. Impact on accountability and legal certainty when a benign and correctly-executed and non-repudiable interaction with the Signer is required . . . 152
6.4.3. Non-interactive public accountability for integrity-protected message (PUB) . . . 153
6.4.4. Relation between non-interactive public accountability and interactive accountability . . . 154
6.4.5. Legal certainty increases with non-interactive public accountability 154
6.4.6. Focus lies on 3^{rd}-party verifiability of integrity and accountability and thus on transferable signatures . . . 155
6.4.7. Level of occurred change detection (*CD) and scope of accountability are related to each other . . . 155
6.5. Scope of block-level and message-level . . . 156
6.6. Proposed extended integrity definition . . . 156
6.7. Extended integrity notion represented by the triple of the three aspects . . . 158
6.8. Existing technical integrity definitions in extended integrity notation using a triple of three aspects . . . 158
6.9. Visualisation of extended integrity notion . . . 159
6.10. Application of the newly proposed definition: Classic digital signature schemes (CDSS) in the extended integrity definition and visualisation 161

7. Technical requirements for integrity and authenticity protection to gain a high probative value 163
7.1. Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer) . . . 164
7.1.1. Requirement 1.1 (Mechanism MUST protect content integrity as requested by the Signer) . . . 165
7.1.2. Requirement 1.2 (Mechanism MUST protect structural integrity as requested by the Signer) . . . 165
7.1.3. Requirement 1.3 (Mechanism MUST protect referential integrity as requested by the Signer) . . . 166
7.2. Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier) . . 166
7.2.1. Requirement 2.1 (Mechanism MUST enable the Signer to control scope and strength of the integrity and authenticity protection) . . . 168
7.2.2. Requirement 2.2 (Mechanism MUST verifiably document Signer's consent for chosen integrity and authenticity protection) . . . 169
7.2.3. Requirement 2.3 (Mechanism SHOULD enable Verifier to verify the proof of consent non-interactively and publicly) . . . 169
7.3. Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification) . . . 170
7.3.1. Requirement 3.1 (Mechanism MUST verify the absence of any subsequent modifications, i.e., ≥1CD integrity) . . . 171
7.3.2. Requirement 3.2 (Mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory) . . 171
7.3.3. Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) . . . 172

7.4. Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) . . . 173
7.4.1. Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) . . . 173
7.4.2. Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer) . . . 175
7.4.3. Notes on Requirement 4 in the multi-Sanitizer settings . . . 176
7.4.4. Link to the legal signatory is out of scope of Requirement 4 . . . 176
7.5. Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate) . . . 177
7.5.1. Key management is part of Requirement 5, but details are out of scope of this thesis . . . 177
7.5.2. Notes on achieving Requirement 5 . . . 178
7.6. Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) . . . 178
7.6.1. Notes on achieving Requirement 6 . . . 179
7.6.2. Relations between Requirement 6 and other requirements and the thesis security assumptions . . . 179
7.7. Requirement 7 (Mechanism MUST achieve a legally accepted state-of-practice strength of the cryptographic algorithm used for integrity and authenticity protection) . . . 179
7.7.1. Requirement 7.1 (Mechanism MUST achieve cryptographic protection against a forgery) . . . 180
7.7.2. Requirement 7.2 (Mechanism MUST be as good as the legally accepted current state of practice for integrity and authenticity protection) . . 182
7.7.3. Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms) . 183
7.8. Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional)) . . . 183
7.9. Relation between technical requirements for increased probative value and analysis results . . . 184
7.10. Relation between technical requirements for an increased probative value and the German description of signature functionalities . . . 187

II Technical requirements for a legally sufficient confidentiality protection for personal data and trade secrets **189**

8. Confidentiality protection by data removal to protect trade secrets and personal data **191**
8.1. Legal permission to protect confidentiality by the sanitization of information in UK and Germany . . . 192
8.1.1. UK FOIA and German IFG: Reasons for redactions in general . . . 193
8.1.2. UK FOIA and German IFG: Notion of redaction and its goals . . . 194
8.1.3. UK FOIA and German IFG: Personal data and trade secret related reasons for redactions . . . 195
8.1.4. EU Regulation 1049/2001: Personal data and trade secret related reasons for withholding information . . . 196

8.2. Analysis of confidentiality requirements in definitions of current (personal) data protection law in US, EU, Germany 197
8.2.1. US law: Privacy definition by *Solove* 197
8.2.2. US law: Confidentiality requirements from US HIPAA, HITECH, and ARRA 197
8.2.3. EU law: Definition of personal data following GDPR (EU Regulation 2016/679) 198
8.2.4. EU law: 'right to be forgotten' and 'erasure' from EU GDPR 200
8.2.5. German law: 'erasure' from BDSG 200
8.2.6. International law: OECD privacy guidelines 201
8.2.7. EU law: Definition of data minimisation principle from EU GDPR . 202
8.2.8. EU law: Directive 45/2001 and Regulation 1049/2001 203
8.3. Legal definition of trade secret protection in UK, EU and international law . 203
8.3.1. UK law: Legal definition of trade secret 204
8.3.2. EU law: Legal definition of trade secret 204
8.3.3. International law: Legal definition of trade secret from WTO 204
8.3.4. EU law: Personal data versus trade secrets in the Directive 95/46/EC 205
8.3.5. EU and German law: No general right to trade secret protection . . 206
8.4. Analysis results of legal definitions regarding confidentiality protection of personal data and trade secrets 206
8.4.1. Analysis Result 14: Definition of personal data 207
8.4.2. Analysis Result 15: Definition of trade secret 207
8.4.3. Analysis Result 16: Legal texts acknowledge the importance of confidentiality protection of personal data and the high-level descriptions of the goal of confidentiality coincide 208
8.4.4. Analysis Result 17: Legal texts demand the confidentiality protection of trade secrets if not outweighed by common interests 209
8.4.5. Analysis Result 18: The importance of or the right to confidentiality protection for data might vary over time 210
8.4.6. Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism 211

9. Analysis of existing technical confidentiality and cryptographic privacy definitions 213
9.1. Definitions of privacy protection in selected existing MSS literature 214
9.1.1. Privacy for RSS by *Miyazaki et al.* 214
9.1.2. Privacy for RSS by *Steinfeld et al.* 215
9.1.3. Privacy for RSS by *Johnson et al.* 215
9.1.4. Privacy for SSS by *Ateniese et al.* 217
9.1.5. Privacy for RSS and SSS by *Brzuska et al.* 218
9.1.6. Privacy for SSS by *Agrawal et al.* 221
9.2. Confidentiality definitions with respect to MSS's privacy protection against recoverability 223
9.2.1. Definition of confidentiality from ISO 2700x and ISO 7498-2 223
9.2.2. Definition of confidentiality from RFC 4949 224
9.2.3. US law: Confidentiality in FIPS 199 and NIST SP 800-122 224
9.2.4. Cryptographic definitions of confidentiality based on indistinguishability 225
9.3. Confidentiality protection levels in technical definition for redaction and data protection 227
9.3.1. ISO 29100 on a framework for privacy 227
9.3.2. ISO 27042 on digital evidence mentions sanitization and redaction . 229
9.3.3. Definition of sanitize / sanitization from ISO 27040 and ISO 27050 . 230
9.3.4. Sanitization of electronic media following NIST SP 800-88 231
9.3.5. Redaction of digital documents following ISO 27038 231

9.4. Analysis results of technical definitions regarding confidentiality protection of personal data and trade secrets 232
9.4.1. Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. 233
9.4.2. Analysis Result 21: The definition of the cryptographic property of standard privacy 233
9.4.3. Analysis Result 22: Leakage through the message is out of scope of MSS properties 233
9.4.4. Analysis Result 23: Cryptographic and legal notion of transparency have an opposite meaning 235

10. Technical requirements for legally compliant privacy protection through sanitization 239
10.1. Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength) 240
10.1.1. Requirement 9.1 (Mechanism MUST achieve a suitable cryptographic protection against the recovery of sanitized or redacted data from the adjusted signature) 241
10.1.2. Requirement 9.2 (Mechanism MUST be as good as the legally accepted current state of practice for protection against unauthorized recovery of sanitized or redacted information) 241
10.1.3. Notes on achieving Requirement 9 243
10.2. Requirement 10 (Mechanism MAY prohibit the identification of additional context information further to Requirement 9 without violating Requirement 4)244
10.3. Linkage between technical and legal requirements for confidentiality and analysis results 244

III Proposed constructions of private redactable and sanitizable signatures with an increased probative value 245

11. State of the Art: Harmonised notation of existing security properties for SSS and RSS 247
11.1. SSS: State of the art 248
11.2. SSS: Harmonised high-level description 255
11.3. SSS: Algorithmic description 256
11.4. SSS: Correctness properties 257
11.4.1. Signing correctness for SSS 258
11.4.2. Sanitizing correctness for SSS 258
11.4.3. Proof correctness of SSS 258
11.4.4. ADM recovery correctness and ease of decomposition for SSS 259
11.5. SSS: Sequential executions of Sanit 259
11.6. SSS: Existing security properties in harmonised notation 260
11.6.1. Unforgeability for SSS 260
11.6.2. Immutability for SSS 261
11.6.3. Standard Privacy for SSS following *Brzuska et al.* 263
11.6.4. Transparency for SSS 265
11.6.5. Unlinkability for SSS 269
11.6.6. Accountability for SSS 270
11.7. SSS: Relations between security properties 272
11.8. RSS: State of the Art 273
11.9. RSS: Harmonised high-level description 283
11.10. RSS: Algorithmic description 284
11.11. RSS: Correctness properties 285
11.12. RSS: Sequential executions of Redact 285

11.13. RSS: Existing security properties in harmonised notation 286
11.13.1. Unforgeability for RSS 286
11.13.2. Immutability for RSS 288
11.13.3. Standard privacy for RSS based on *Brzuska et al.* 288
11.13.4. Transparency for RSS 290
11.14. RSS: Relations between RSS security properties 291

12. Shortcomings of state-of-the-art SSS and RSS with respect to the requirements 293
12.1. Shortcoming 1: Not all schemes achieve sophisticated levels of privacy . . . 294
12.1.1. Problems of scheme by *Idalino et al.* towards privacy 295
12.1.2. Problems of *Izu et al.* towards standard privacy 296
12.1.3. Problem of *Wu et al.* towards standard privacy 296
12.1.4. Problems of *Kundu et al.* towards privacy 296
12.1.5. Stronger notions of privacy 298
12.2. Shortcoming 2: No full protection of all aspects of integrity 298
12.2.1. Problems of *Kundu and Bertino* towards unforgeability and structural integrity 298
12.2.2. Problems of *Miyazaki et al.* towards structural integrity 301
12.3. Shortcoming 3: No, or no explicit public form of accountability 301
12.4. Shortcoming 4: No accountability on a the fine-grained scope of blocks, only on the scope of the message 303
12.5. Shortcoming 5: Limited control of the structural integrity protection 304
12.5.1. Problems and limits on structural protection and missing flexibility for non-leaf node redaction 304
12.5.2. Motivation for structure becoming individually redactable 305
12.6. Shortcoming 6: No explicit dual operation to redact, i.e., no explicit 'merge' of two linkable modified documents 306

13. Proposed private SSS with an increased probative value 309
13.1. Proposed SSS property: Non-interactive public accountability (PUB) 310
13.1.1. Shortcoming 3 addressed by non-interactive public accountability . 310
13.1.2. Definition of non-interactive public accountability (PUB) on the block- and message-level 310
13.1.3. Relation between non-interactive public accountability and transparency 313
13.1.4. Relation between non-interactive public accountability and privacy 315
13.1.5. Relation between block-level and message-level accountability properties 315
13.1.6. Visualisation of Non-interactive Public Accountability 317
13.2. Proposed SSS property: Accountability (transparent or non-interactive public) on the level of groups of blocks 319
13.2.1. Shortcoming 4 addressed by block-level accountability 319
13.2.2. Definition of block-level and group-of-blocks-level accountability . . 319
13.2.3. Group-level non-interactive public (PUB) accountability for SSS . . 321
13.2.4. Group-level interactive non-public (INT / transparent) accountability for SSS 322
13.2.5. Relation to existing properties 323
13.2.6. Visual of group-level accountability 324
13.3. Proposed SSS property: Strengthened unlinkability 325
13.3.1. Shortcoming 1 and Shortcoming 3 addressed by strengthened unlinkability 326
13.3.2. Definition of strengthened unlinkability 326
13.3.3. Relation to existing properties 327
13.4. *pubacc*$\mathcal{SSS}$ concept: (Weak | standard) (block-level | message-level) non-interactive public accountability 329
13.4.1. *pubacc*$\mathcal{SSS}$ extended security properties 329
13.4.2. Adjustment of privacy definition for block-level and group-level *pubacc*$\mathcal{SSS}$ 329
13.4.3. *pubacc*$\mathcal{SSS}$ algorithmic description 331

13.4.4. $pubacc\mathcal{SSS}$ correctness properties 332

13.5. $pubacc\mathcal{SSS}$ cryptographic instantiation 332

13.5.1. $pubacc\mathcal{SSS}$ constructions 332

13.5.2. $pubacc\mathcal{SSS}$ and $pubacc\mathcal{SSS}^{Block}$ complexity and runtime 336

13.5.3. $pubacc\mathcal{SSS}$ and $pubacc\mathcal{SSS}^{Block}$ security proof 336

13.5.4. $pubacc\mathcal{SSS}$ and $pubacc\mathcal{SSS}^{Block}$ integrity offer and visualisation 338

13.6. $blockacc\mathcal{SSS}$ Concept: Non-Interactive public accountability for groups of block(s) 340

13.6.1. $blockacc\mathcal{SSS}$ extended security properties 340

13.6.2. $blockacc\mathcal{SSS}$ algorithmic description 340

13.6.3. $blockacc\mathcal{SSS}$ correctness properties 341

13.7. $blockacc\mathcal{SSS}$ cryptographic instantiation 341

13.7.1. $blockacc\mathcal{SSS}$ cryptographic preliminaries 341

13.7.2. $blockacc\mathcal{SSS}$ constructions 343

13.7.3. $blockacc\mathcal{SSS}$ runtime and complexity 346

13.7.4. $blockacc\mathcal{SSS}$ security proof 346

13.7.5. $blockacc\mathcal{SSS}^{INT}$ and $blockacc\mathcal{SSS}^{PUB}$ integrity offer and visualisation349

13.8. $unlinkable\mathcal{SSS}$ concept: Strong unlinkability (privacy) and non-interactive public accountability for the message 352

13.8.1. $unlinkable\mathcal{SSS}$ extended security properties 352

13.8.2. $unlinkable\mathcal{SSS}$ algorithmic description 352

13.8.3. $unlinkable\mathcal{SSS}$ correctness properties 353

13.9. $unlinkable\mathcal{SSS}$ cryptographic instantiation 354

13.9.1. Cryptographic preliminaries 354

13.9.2. $unlinkable\mathcal{SSS}$ construction 354

13.9.3. $unlinkable\mathcal{SSS}$ runtime and complexity 355

13.9.4. $unlinkable\mathcal{SSS}$ security proof 356

13.9.5. $unlinkable\mathcal{SSS}$ integrity offer and visualisation 357

13.10. Implementation of an SSS on a secure signature creation device (smart card) 357

13.10.1. Secure smart card test environment (same for RSS and SSS) 357

13.10.2. $pubacc\mathcal{SSS}^{Block}$ on a smart card 359

13.10.3. Secure smart card implementation overview 359

13.10.4. Performance evaluation of $pubacc\mathcal{SSS}^{Block}$ and other schemes on a secure smart card 361

14. Proposed private RSS with an increased probative value 363

14.1. Proposed RSS property: Accountability and accountable redactable signature scheme (ARSS) 364

14.1.1. Signer-Accountability for an RSS offering non-interactive public accountability 364

14.1.2. Goal of ARSS: Make RSS Sanitizer- and Signer-accountable and offer non-interactive public accountability ($\geq$1CD$-$PUB) 365

14.1.3. Shortcoming 3 addressed by ARSS 365

14.1.4. ARSS algorithmic description 366

14.1.5. ARSS correctness properties 367

14.1.6. ARSS extended security model 367

14.1.7. Visualisation of ARSS 368

14.2. Proposed RSS property: Mergeability, the explicit inverse of redact (MRSS) 369

14.2.1. Shortcoming 6 addressed by MRSS 369

14.2.2. MRSS algorithmic description 370

14.2.3. MRSS correctness 370

14.2.4. MRSS extended security model 371

14.2.5. Relation between MRSS properties and other security properties 374

14.3. Proposed RSS property: Unlinkability 376

14.4. Proposed RSS property: Linkability through explicit mergeability of 'variants' of redacted documents 377

14.5. $privRSS$ concept: Transparent and standard privacy for tree-based documents with intermediary node redaction . . . 377
14.5.1. $privRSS$ extended security properties . . . 378
14.5.2. $privRSS$ algorithmic description for an RSS for trees (RSS_T) . . . 380
14.5.3. $privRSS$ correctness properties . . . 381
14.6. $privRSS$ cryptographic instantiation . . . 381
14.6.1. Cryptographic preliminaries . . . 381
14.6.2. $privRSS$ construction . . . 382
14.6.3. $privRSS$ runtime and complexity . . . 384
14.6.4. $privRSS$ security proof . . . 385
14.6.5. $privRSS$ integrity offer and visualisation . . . 386
14.7. $flexRSS$ concept: RSS for trees with flexible intermediate node redaction with Signer control . . . 387
14.7.1. $flexRSS$ extended security properties . . . 388
14.7.2. $flexRSS$ algorithmic description . . . 389
14.7.3. $flexRSS$ correctness properties . . . 390
14.8. $flexRSS$ cryptographic instantiation . . . 390
14.8.1. Cryptographic preliminaries . . . 391
14.8.2. $flexRSS$ construction . . . 393
14.8.3. $flexRSS$ runtime and complexity . . . 395
14.8.4. $flexRSS$ security proof . . . 395
14.8.5. $flexRSS$ integrity offering and visualisation . . . 397
14.9. $structRSS$ concept: RSS allowing private redaction of structure independent from content . . . 399
14.9.1. Shortcoming 5 addressed by $structRSS$ . . . 399
14.9.2. $structRSS$ extended security model . . . 399
14.9.3. $structRSS$ algorithmic definition . . . 400
14.9.4. $structRSS$ correctness . . . 401
14.10. $structRSS$ cryptographic instantiation . . . 401
14.10.1. Cryptographic preliminaries . . . 401
14.10.2. $structRSS$ construction . . . 403
14.10.3. $structRSS$ runtime and complexity . . . 405
14.10.4. $structRSS$ security proof . . . 406
14.10.5. $structRSS$ integrity offer and visualisation . . . 408
14.11. $mergeRSS$ Concept: Allow merging several redacted documents into one, if and only if they stem from the same signed source . . . 409
14.11.1. $mergeRSS$ extended security properties . . . 409
14.11.2. $mergeRSS$ algorithmic description . . . 410
14.12. $mergeRSS$ cryptographic instantiation . . . 410
14.12.1. Cryptographic preliminaries . . . 410
14.12.2. $mergeRSS$ construction . . . 412
14.12.3. $mergeRSS$ runtime and complexity . . . 414
14.12.4. $mergeRSS$ security proof . . . 414
14.12.5. $mergeRSS$ integrity offer and visualisation . . . 416
14.13. $pubaccRSS$ Concept: Make the redact operation key dependent to gain accountability . . . 417
14.13.1. $pubaccRSS$ extended security properties . . . 417
14.13.2. $pubaccRSS$ algorithm . . . 417
14.13.3. $pubaccRSS$ correctness properties . . . 417
14.14. $pubaccRSS$ cryptographic instantiation . . . 417
14.14.1. $pubaccRSS$ construction . . . 418
14.14.2. $pubaccRSS$ runtime and complexity . . . 418
14.14.3. $pubaccRSS$ security proof . . . 419
14.14.4. $pubaccRSS$ integrity offer and visualisation . . . 423
14.15. Implementation of an RSS on a secure signature creation device (smart card) 424
14.15.1. Smart card test environment . . . 424
14.15.2. $mergeRSS$ on smart card . . . 424
14.15.3. Construction of $mergeRSS$ on smart card . . . 427

14.15.4. Security properties of the smart card version of *mergeRSS* 428
14.15.5. Smart card implementation overview 428
14.15.6. Performance evaluation of *mergeRSS* on a smart card 430

15. Separating secure RSS from secure SSS 431
15.1. Separation of security models 432
15.1.1. The property of unforgeability remains unchanged 432
15.1.2. The property of weak immutability for SSS is introduced 432
15.1.3. The property of strengthened privacy for SSS is introduced 433
15.1.4. The property of weak privacy for RSS is introduced 434
15.1.5. The property of strong privacy for RSS is introduced 435
15.1.6. The property of weak block-level non-interactive public accountability for SSS is introduced 436
15.1.7. Implications between new properties and resulting separation of SSS and RSS 436
15.2. Generic transformation of an SSS into an RSS 437
15.3. Minimum requirements and limitations to transform an SSS into an RSS . . 439

IV Related and previous work, evaluation, applications and conclusion 443

16. Previous and related work on probative value or data protection requirements of 'non-standard' digital signatures 445
16.1. Related work in the area of legal analysis of other signature schemes 445
16.1.1. Related work on the container signature, e.g., by *Gennen* from 2009 445
16.1.2. Related work on mobile signatures by *Orthacker et al.* from 2010 and by *H. Rossnagel* from 2004 446
16.1.3. Related work on identity based signature schemes by *Sorge* from 2014 448
16.1.4. De-Mail law in Germany legally assigns a certain technical mechanism a high probative value 448
16.2. Previous and related work in the area of legal analysis of MSS 449
16.2.1. Related work by *Stranacher, Krnjic, Zwattendorfer, and Zefferer* [460] from 2013 450
16.2.2. Related work by *Stranacher, Krnjic, and Zefferer* [459] from 2013 . . . 451
16.2.3. Related work by *Zwattendorfer and Slamanig* [513] from 2013 451
16.3. No related work existed on the legal implications of MSS 452
16.4. Impact on the state of the art 452

17. Evaluation of the data protection capabilities and the probative value achieved by the proposed RSS and SSS 455
17.1. Evaluation of data protection capabilities of proposed schemes (Req. 9) . . . 457
17.1.1. MSS uphold the integrity protection, which is beneficial 457
17.1.2. Private MSS are a technical tool to minimise data, which is beneficial 458
17.1.3. Data protection requirement: Achieve at least cryptographic standard privacy following *Brzuska et al.* 458
17.2. Evaluation Result 1: All proposed schemes from Chapters 13 and 14 fulfil data protection requirement (Req. 9) as they achieve standard privacy 460
17.3. Evaluation of probative value of all schemes (Req. 1–7) 461
17.3.1. Fulfilment of Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer): Integrity is protected to the signatory's policy 462
17.3.2. Fulfilment of Requirement 2: Signatory's consent is verifiable 463
17.3.3. Fulfilment of Requirement 3: Any subsequent modification is detected ($\geq$1CD), either INT or PUB 465
17.3.4. Fulfilment of Requirement 4: Based on detection of any occurred subsequent modification ($\geq$1CD) achieve Signer accountability . . . 466
17.3.5. Fulfilment of Requirement 5: Linkage towards the legal signatory . 468
17.3.6. Fulfilment of Requirement 6: Run secret-key-relevant operations inside QSCD/SSCD 469

17.3.7. Fulfilment of Requirement 7: Provably secure algorithms 469
17.4. Evaluation Result 2: MSS are 'elementary' electronic signatures following Regulation 910/2014 470
17.5. Comparison of signature verification outcomes of an MSS and a qualified electronic signature generated by a CDSS 471
17.5.1. Two cases after verifying a CDSS with NCA – 1CD integrity 471
17.5.2. Five cases after verifying an MSS with ACA – ≥1CD integrity and 3^{rd}-party verifiability 473
17.5.3. Additional cases due to block-level properties 478
17.6. Evaluation Result 3: A signature by an MSS with ACA – ≥1CD – PUB, that yields that the signed document is unmodified, is a qualified electronic signature following Regulation 910/2014 479
17.7. Discussion of possible doubts against evidence created by MSS 480
17.7.1. Doubt: Algorithm not acceptably strong for a qualified electronic signature 481
17.7.2. Doubt: Algorithm is not accredited 481
17.7.3. Doubt: Keys not managed securely 482
17.7.4. Doubt: Keys not secret 482
17.8. Discussion of impact on general limitations of this evaluation 483
17.8.1. General limitation: The thesis analysed the low levels of cryptographic properties of RSS and SSS; its not a full expert opinion . . 483
17.8.2. General limitation: Direct representation even by a secure malleable signature schemes is not always allowed 483
17.9. Evaluation Result 4: A signature with an MSS with ACA – ≥1CD –PUB integrity but no Sanitizer-accountability generates a legal blanket statement 484
17.10. Evaluation Result 5: Rules to award evidence with a legal presumption of authenticity carry over to MSS 485
17.11. Evaluation Result 6: MSS can offer detectability of double sanitized documents created by sanitization or redaction from the same origin, but no ordering can be derived 486
17.12. Evaluation Result 7: MSS can be facilitated to prove that a different (or original) version of a sanitized document exists, assuming one is in possession of that version's message-signature pair 487

18. Applications empowered by private RSS and SSS with an increased probative value 489
18.1. Full system integration: Sanitizable waybills and laboratory reports for increased food safety in food supply chains 490
18.1.1. The supply chain model with signature related control points 491
18.1.2. State of signatures 493
18.1.3. SSS as XML DSIG compatible signature 494
18.1.4. Waybill protection and subsequent authorized modification within ReSCUeIT 496
18.1.5. Implementation of $pubacc\mathcal{SSS}$ as web services for generation, verification, sanitization of signatures in XML within ReSCUeIT 500
18.2. Proof of concept: Information source authentication and integrity preserving application of privacy-enhancing technology (PET) in the Internet-Of-Things (IoT) for healthcare 501
18.2.1. RSS on IoT devices gathering healthcare data 502
18.2.2. Implementation and evaluation on three platforms 503
18.2.3. Evaluation and observations 505
18.3. Concept: Information source authentication and integrity preserving application of privacy-enhancing technology (PET) in the smart energy grid 506
18.3.1. Using an RSS to control privacy-enhancing integrity violations of energy meter output 507
18.3.2. System description and integrity requirements 508
18.3.3. Privacy threats in the smart grid domain 508
18.3.4. Perturbation as a privacy-enhancing technology 509

18.3.5. RSS with high probative value to control the PET's invasiveness and the resulting data quality . . . 509
18.3.6. Security and privacy properties gained with the RSS . . . 511
18.3.7. Implications of RSS signatures with an increased probative value . 512
18.4. Concept: RSS for product content documentation of food supply chains . . 512
18.5. Concept: Exercising data governance rights and give legal consent to flow of data . . . 514
18.5.1. Identified problems . . . 515
18.5.2. Example of a data subject's request in a simple composed service . 516
18.5.3. Application of an SSS to guarantee authenticity of the response data while allowing to adapt responses to keep trade-secrets . . . 517
18.6. Concept: Verifiable and transparent database unions of data distributed in the cloud. . . . 519
18.7. Concept: Electronic blanket statements . . . 521
18.8. Definitional: Contingency – the dual of integrity . . . 522
18.8.1. Motivating example for contingency in information and communication technology (ICT) . . . 522
18.8.2. Digital signatures protection of integrity . . . 522
18.8.3. Levels of weakened integrity leading up to contingency . . . 523
18.8.4. A definition of contingency or intervenability . . . 524
18.8.5. Applications and legal implications . . . 525
18.8.6. Towards constructions for contingency protection . . . 526
18.8.7. Construction of a contingent signature scheme using an SSS . . . 527
18.8.8. Security of the contingency construction . . . 528
18.8.9. Accountability and integrity mixed with contingency . . . 529

19. Conclusion, contributions and future work 531
19.1. Conclusion . . . 531
19.2. Main results and contributions of this thesis . . . 532
19.2.1. Harmonisation of existing cryptographic properties and design of new or refined properties . . . 532
19.2.2. Extended integrity notion that captures important differences between RSS, SSS and CDSS and within each class . . . 533
19.2.3. Formal separation of SSS and RSS . . . 534
19.2.4. Alignment of CDSS and MSS by proposing the property of non-interactive public accountability (PUB) and ≥1CD integrity . . . 535
19.2.5. Legal evaluation yields that MSS can award an increased probative value . . . 536
19.2.6. New RSS and SSS constructions provide standard privacy and an increased probative value . . . 537
19.2.7. SSS technically protect the dual of integrity: Contingency . . . 539
19.3. Future work and open problems . . . 539

20. Bibliography XXIII

A. Record of Publications LIII
A.1. Peer-reviewed articles in journals related to this thesis . . . LIII
A.2. Peer-reviewed articles in conference proceedings related to this thesis . . . LIV
A.3. Technical reports and project deliverables related to this thesis . . . LIX
A.4. Other peer-reviewed publications . . . LX

B. Glossary of Terms and Acronyms LXV

C. List of Figures LXXI

D. List of Tables LXXVII

1 — Introduction, motivation, research question, scope, methodology and organisation

1.1. Introduction

The amount of disputes over digitally exchanged documents increases as the amount of digital information increases[2]. These documents are referred to as 'electronic documents' in European Union (EU) legal terminology. The burden of production of evidence[3] is on one of the conflicting parties when electronic documents are taken into court as evidence to reach for a settlement. This is one of the reasons why economically valuable or otherwise important transactions include the generation of some form of documents, e.g., a notice of receipt gets signed by the recipient. The documents can later serve as evidence toward a claim that an action has indeed occurred as claimed. Equally, digital or electronic documents must provide businesses and individuals with a credible source for evidence.

The protection of privacy on the other hand calls for data minimisations: This might require to only release necessary information — a subset — from a credible document. Documents often contain more information than needed; For example when a document was created at a time before the request for a partial release, or when the document was created for another, more general purpose. The objective of this thesis is to provide cryptographic mechanisms that (1) protect an electronic document with the necessary strength to become a source of evidence with a high probative value and allow to remove unnecessary information such that (2) the removed information remains private while (3) the remaining information shall — at best — still be considered as a source of evidence.

Probative value measures the legal impact. This thesis is interested in the increase of probative value achieved by a technical mechanism if an electronic document is protected by it. Evidence, which possesses a *high* probative value, is useful as it allows to convince the judge(s) quicker. For example, so-called 'documentary evidence' has a high probative value: A case in civil proceedings in Germany in which only such evidence is presented to the court follows a distinct set of rules and is usually decided faster due to limited possibilities to oppose the evidence[4]. The direct and indirect costs of the trial can be reduced by the reduced time such a trial takes. The regulatory domain of the EU is the legal focus of this thesis. Within the EU the probative value of electronic documents is assigned by national legal texts based on technical requirements set-forth by European-wide legislations[5]. The legal texts state that once an electronic document is electronically signed, its legal probative value can be increased and it might become documentary evidence. However, the technical signature method used must fulfil specific requirements to have a pre-defined legal impact. Specific legal texts and technical standards indicate that certain algorithms are technically capable of fulfilling the legal requirements, hence they are referred to as legally recognised. Using a legally recognised mechanism will increase what is often referred to as *legal certainty*. This thesis interprets legal certainty to mean the following[6] in the light of a technical signature mechanism: It is known in advance with sufficient certainty that the court will assign a high probative value to that electronically signed document, due to the circumstance that the respective technical signature mechanism was used. This thesis carried out a requirement elicitation process in order

[2] Cmp. `https://www.domo.com/blog/2014/04/data-never-sleeps-2-0/` and `https://www.domo.com/blog/2015/08/data-never-sleeps-5/` [last accessed: Jan. 2018]

[3] Please see the Glossary in Appendix B for German-English translations of legal terms used in this thesis.

[4] Following §§ 592 ff. ZPO n.F.; in German 'Urkundenprozess'.

[5] Relevant on EU level used to be the Directive 1999/93/EC, which was superseded by Regulation 910/2014 in mid-2016.

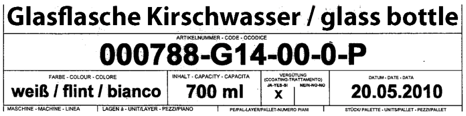
[6] See also Definition 111 in Sec. 6.4.2 on page 152.

to design new cryptographic mechanisms such that they can provide legal certainty. Hence, the benefit of a high probative value is also highlighted by the increased economic value of such an electronic document: Knowing that a certain electronic document has most likely an increased probative value might even allow to settle disputes without the need for a trial in court.

Integrity and accountability are technical terms related to evidence: ISO 27000 [245] defines five technical protection goals of information security: confidentiality, integrity, availability, accountability, and auditability [245]. A high probative value requires two out of the five: *integrity* and *accountability*. Protecting integrity allows to verify that information was not modified in any unauthorized way after the protection was applied. Accountability allows appointing the party from which information originated.

Classic digital signature schemes (CDSS) are technically well-known and legally recognised cryptographic mechanisms to achieve — flanked by organisational methods — the integrity and accountability that introduce a high probative value. When a legally recognised digital signature scheme is used correctly, it establishes the legally necessary protection and thus the signed document's probative value is increased as its legal impact is increased. This then can help to increase also the economic value of the signed document.

Legal goals like data protection and privacy are considered as legal applications of the technical aspect of *confidentiality*. Additionally to integrity and accountability this thesis designs new signature mechanisms — and analyses existing ones — that achieve a cryptographic property termed *privacy*. The aim of the cryptographic property of privacy, as used in this thesis, is best described with by the 'removal' of information. For example the information `000788-G14-00-0-P` contained in Fig. 1 has been cloaked/blackened/removed/redacted/sanitized[7] to create Fig. 2.



Glasflasche Kirschwasser / glass bottle

ARTIKELNUMMER - CODE - OCODICE
000788-G14-00-0-P

FARBE - COLOUR - COLORE
weiß / flint / bianco

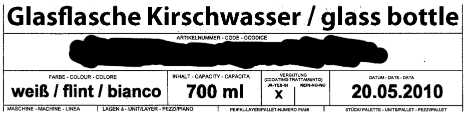
INHALT - CAPACITY - CAPACITA
700 ml

VERGÜTUNG (CCOATING-TRATTAMENTO)
JA-YES-SI x
NEIN-NO-NO

DATUM - DATE - DATA
20.05.2010

MASCHINE - MACHINE - LINEA | LAGEN à - UNIT/LAYER - PEZZI/PIANO | PE/PAL-LAYER/PALLET-NUMERO PIANI | STÜCK/ PALETTE - UNITS/PALLET - PEZZI/PALLET

Figure 1. Snippet from a supply chain document containing the original information



Glasflasche Kirschwasser / glass bottle

ARTIKELNUMMER - CODE - OCODICE
[redacted]

FARBE - COLOUR - COLORE
weiß / flint / bianco

INHALT - CAPACITY - CAPACITA
700 ml

VERGÜTUNG (CCOATING-TRATTAMENTO)
JA-YES-SI x
NEIN-NO-NO

DATUM - DATE - DATA
20.05.2010

MASCHINE - MACHINE - LINEA | LAGEN à - UNIT/LAYER - PEZZI/PIANO | PE/PAL-LAYER/PALLET-NUMERO PIANI | STÜCK/ PALETTE - UNITS/PALLET - PEZZI/PALLET

Figure 2. Snippet where information got cloaked/blackened/removed/redacted/sanitized

Henceforth, the words cloaked/blackened/deleted/removed/redacted/sanitized[7] are used to describe that some original information has been transformed such that the previous state can no longer be retrieved/recovered//extracted/restored from the information after it was transformed. The term 'original' is used to refer to the non-cloaked or non-sanitized information. A cryptographic mechanism that achieves privacy can be used as a so-called privacy-enhancing technology (PET) when information disclosure must be minimised, e.g., to follow the data minimisation principle mentioned in data protection legislation [91, 174, 176].

The objective of this thesis is to achieve both goals in conjunction: First, to cryptographically protect integrity and enable accountability to increase the probative value. Second, to allow cryptographically provable data protection by allowing subsequent modifications to take place without the possibility to re-gain any information after it was cloaked. Consequently, the proposed cryptographic signature mechanisms are designed to fulfil some or all of the legal requirements — gathered in the light of subsequent authorized modifications. As a result, the electronically signed document gets awarded with a legally high probative value and still enables subsequent authorized modifications in order to be compliant with data protection regulation. On the surface, the goals appear to be contradicting: On the one hand there is the need to protect information against unauthorized modifications that is inherent in the protection goal of integrity and legal accountability (1^{st} goal) and on the other hand there is the need for authorized subsequent modifications for privacy (2^{nd} goal). *Miyazaki, Susaki, Iwamura, Matsumoto, Sasaki, and Yoshiura* [346] termed this the *digital document sanitizing[7] problem* in 2003[8]. More details follow shortly in Sec. 1.2.

[7] The American English spelling from original work is retained for terms and quotes.

[8] It first appeared as a technical report in Japanese in 2003 [346] in the English title, a paper written in English language with a matching title appeared in 2005 [347].

The existing solution proposed in the literature is a class of cryptographic mechanisms known as *private malleable signature scheme (private MSS)*. A malleable signature scheme is a specialised digital signature scheme with additional cryptographic properties setting them apart from the classic digital signature. On a very abstract level, the MSS adds the functionality to authorize subsequent modifications within a predefined scope by a predefined party while keeping the properties of digital signatures. Before and after subsequent authorized modifications an MSS allows in particular all of the following: (1) to detect the presence of any unauthorized change, (2) it allows to authenticate the origin via the signer's verification key, and (3) it provides non-repudiation of origin. If the MSS is cryptographically *private*, the subsequent modification(s) that cloak certain information in an authorized manner keep the original information confidential. The concept of MSS has been proposed in several ways in a number of research publications: The early works date back to *Merkle*'s hash tree [338] in 1989 and chameleon hashing proposed by *Krawczyk and Rabin* [292] in 2000. Works describing the use for document sanitization are redactable signature schemes (RSS) described by *Johnson, Molnar, Song, and Wagner* [273] and *Steinfeld, Bull, and Zheng* [455] in 2002. And sanitizable signature schemes (SSS) introduced by *Ateniese et al.* [12] in 2005. For the state of the art see Sec. 11.1 and Sec. 11.8; A quick introduction is given soon in Sec. 1.2.2.

The research question of this thesis is as follows:

> Can a malleable signature scheme be private to be compliant with EU data protection regulation **and at the same time** fulfil the integrity protection legally required in the EU to achieve a high probative value for the data signed?

The research question was motivated by the interesting duality of the goals achieved by MSS and by the finding that — despite this duality giving rise to manifold applications — MSS are not in widespread use. Further, their legal status was unknown prior to this thesis: No analysis of their probative value existed, not even a discussion of MSS's cryptographic properties in the light of legal requirements was known.

1.2. Motivation, problem statement and examples

The problem that arises from the combination of, on the one hand, authorising subsequent modifications and, on the other hand, retaining a verifiably digital signature, was captured by *Miyazaki et al.* as the **digital document sanitizing[9] problem** [344, 346, 347]:

> "**Appropriate alteration of** some **signed documents**, however, should be allowed because there are security requirements other than that for the integrity of the document. In the disclosure of official information, for example, sensitive information such as personal information or national secrets is masked when an official document is sanitized so that its nonsensitive information can be disclosed when it is demanded by a citizen. If this disclosure is done digitally by **using the current digital signature schemes, the citizen cannot verify the disclosed information correctly because the information has been altered to prevent the leakage of sensitive information**. That is, with current digital signature schemes, the confidentiality of official information is incompatible with the integrity of that information." [10] [344, 347]

1.2.1. Problem statement

The literature lists many applications that benefit from a solution to the digital document sanitization problem: sanitizing medical records, secure routing, blank signatures[11], or collaborative document generation [12, 65, 222, 224, 463]. However, to really become first class citizens in the world of practically and legally usable electronic signatures the acceptance and integration barriers must also be low. Thus,

[9] The American English spelling from original work is retained for terms and quotes.

[10] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[11] The term 'blank signatures' is from the title of [224], it describes an underspecified signed form, i.e. a signed bank cheque where the amount is made out in blank; this legal term is called 'blanket statement' in thesis and discussed in more detail in Sec. 4.5.5.

this thesis complements the digital document sanitization problem with three important additional requirements: First and second are legal requirements from two domains: electronic signature and data protection. The third requirement is practicality: flexibility, tolerable overhead, the ability to re-use existing infrastructures.[12]

Combined this results in the following research problem:

Research problem: Can one design a provably secure[a] cryptographic scheme that

- allows to authorize subsequent modifications to an integrity-protected document such that on a document which has undergone only authorized modifications, the origin can still be authenticated to be the same as the original's and the absence of unauthorized modifications can be established[b],
- **and** achieves the highest legally possible probative value following EU electronic signature legislation (for an original as well as an authorized modified document),
- **and** achieves a cryptographic level of confidentiality for cloaked information to be compliant with data protection legislation in the EU,
- **and** offers the flexibility, performance and integratability needed for a broad applicability in many domains?

[a] The term 'provably secure' is used in this thesis to indicate that the security is established following the "[...] rigorous analysis methodology of cryptography (which evolves around clear definitions and rigorous inference rules) [...]" [207]. It shall further highlight that security is only established against that clearly defined goal. *Goldreich* criticised in that same essay [207], from which the previous quote is taken, that "[...] within the domain of a rigorous analysis of cryptography, the term "provable security" (and in general "provable property") makes no sense; that is, the adjective "provable" adds nothing to the claim of security (assuming that the claim is valid, and it cannot be applied if the claim is invalid or unknown to be valid)." [207]. The author of this thesis indeed thinks that *Goldreich* is correct in that observation, still the notion was found in works in the area of MSS and, in order to make a clear statement that such a cryptographically rigorous proof is offered, the term was chosen. For more discussion see Sec. 2.3.

[b] This is the digital document sanitizing problem re-phrased.

1.2.2. Malleable signature schemes with added goals

The concept of a malleable signature scheme (MSS) has been introduced to solve the digital document sanitizing problem.[13] As a solution they allow a semi-trusted party to subsequently modify a signed document. Thus, the signing party gives up some control over the contents of the signed document and hence delegates the right to the semi-trusted party being able to produce the "appropriate alteration" [347] bearing a signature which still verifies and still relates to the signing party. The delegatee does not receive the full rights, only pre-defined subsequent modifications are allowed, hence the notion of being semi-trusted.

Secure private malleable signature schemes achieve the cryptographic property of unforgeability, that in turn allows the schemes to achieve integrity protection, and accountability in combination with a cryptographically sufficient level of confidentiality for cloaked information that allows to achieve a property called privacy.

The subsequently modified document can still be related to the Signer if only authorized modifications were applied by an authorized Sanitizer. Regarding the integrity and accountability, the core idea behind an MSS is to enable a level of integrity protection that would allow a third party, called Verifier, by means of signature verification to identify the signing party, called the Signer, as the creator of the signature for that document. Still, the MSS detects unauthorized modifications.

[12] One — if not the most — important infrastructure in the area of electronic signatures is the public key infrastructure (PKI) and the related legally recognised certification authorities (CAs) also know as trust service providers under Europe's latest signature legislation (Regulation 910/2014, or eIDAS for short).

[13] Regarding redactable signatures, "[i]n other words, digital signatures and redactions are made compatible with this technology." [230]; the Hitachi online glossary [230] is used as a source here because it summarises and motivates the functionality and because *Kunihiko Miyazaki*, the author and co-author of many RSS papers [222, 344, 346–348], works/worked at Hitachi [343].

This resembles the accountability and integrity protection that digital signature schemes usually achieve. However, contrary to a classic digital signature scheme, an MSS allows to authorize a third party, called the Sanitizer, to modify the integrity-protected document and at the same time to derive an adjusted signature such that the adjusted signature still verifies for the subsequently modified document. Thus, the subsequently modified document can still be related to the Signer via its signature verification key.

This thesis definition of a MSS, given in Sec. 3.4.1, is as follows:

Definition 45 : Malleable Signature Scheme (High-Level)

> *A malleable signature scheme is a secure digital signature scheme that*
>
> - *allows the Signer to authorize one (or more) semi-trusted entities, called Sanitizer, to carry out designated modifications to certain designated blocks of a document and produce a valid signature of the resulting authorized modified document;*
> - *allows a Sanitizer to produce the valid signature on the authorized modified document, which can be produced without interaction with the original Signer;*
> - *prohibits even a designated Sanitizer, let alone a third party, to produce a valid signature for any unauthorized modification;*
> - *hence the Sanitizer must only be semi-trusted.*

The Sanitizer is not required to interact with the Signer for a successful authorized modification and the adjustment of the signature. Especially, an MSS does not require any re-application of the integrity protection over the modified document by the Signer. Most notably, the Sanitizer does not need to know the Signer's secret signature generation key to carry out an authorized modification.

A private malleable signature scheme also protects confidentiality of original information. Namely, the MSS keeps the original version confidential after the signed document has been subsequently modified. Henceforth, this property is termed privacy — in line with the existing literature in the realm of MSS.

Therefore, any allowed subsequent modification, i.e., the authorized removal or change of some blocks from the signed document, must be done such that the previous information remains confidential. Hence, a Verifier who will not be given access to the original blocks anymore, will not be able to gain any information about them. For example, from a sanitized medical record, one cannot retrieve any additional information besides what is given. Thus, if the malleable signature scheme achieves the cryptographic property of privacy, it allows application domains like removing certain information from medical documents [12].

The 1^{st} goal is to achieve a high probative value when an electronic document is electronically signed with a malleable signature scheme. Its legal probative value is increased only if the technical signature method used fulfils specific requirements set forth by electronic signature legislation. Henceforth, the technical mechanism which is used to create electronic signatures is called a digital signature scheme. The produced output is a digital signature. A digital signature protects the integrity and provides authentication of origin of data, non-repudiation and with this allows establishing accountability for the signed data. The security goals of integrity, accountability and non-repudiation have been technically defined as follows by ISO 7498 [239][14] and ISO 13888 [243][15]:

> "[Data integrity is the] property that data has not been altered or destroyed in an unauthorized[16] manner" [239].
>
> "[Accountability is the] property that ensures that the actions of an entity may be traced uniquely to the entity" [243].
>
> "[Non-repudiation of origin is a] service intended to protect against the originator's false denial of having created the content of a message and of having sent a message" [243].

[14] Data origin authentication as defined in ISO 7498-2 is discussed in Sec. 5.2.1.

[15] Non-repudiation defined in ISO 13888 is discussed in Sec. 5.2.5.

[16] The American English spelling from original work is retained for terms and quotes.

The term non-repudiation is meant technically, not legally. Integrity, accountability as well as non-repudiation of origin are technical goals. With means of cryptographic primitives technical evidence is generated. It must be differentiated from the legal possibilities to repudiate actions.

Note that technical evidence generated by sufficiently strong cryptographic schemes cannot be cryptographically repudiated, but of course it could still be repudiated in court, which will make the ultimate decision in a dispute involving the cryptographically protected electronic document[17] . However, solid technical evidence can be used to increase the probative value. This thesis discusses the technical notion of non-repudiation in more detail in relation to accountability in Sec. 2.10.2 and several selected technical definitions are analysed in the light of authorized subsequent modifications in Subsections 5.2.3 to 5.2.6.

Technically, digital signature schemes allow parties in the role of the Verifier to detect any *subsequent modification* done by any other party than the Signer which occurred after[18] the signature's generation by the Signer. Digital signature schemes that fulfil this are said to be **unforgeable**. Cryptography offers digital signature schemes that can be and are in practice used as electronic signatures, such as RSASSA-PSS from PKCS-v2.2 [422], which is legally recognised [74]. This thesis will refer to those schemes as *classic* digital signature schemes (CDSS). For the positive verification of a CDSS the Verifier needs to reproduce the exact document m as it was signed, i.e., from the first bit to its last bit, the Signer's signature σ_m over m, and additionally, the Signer's signature verification key $\mathsf{pk}_{\mathsf{sig}}$. Simplified, European legal texts on electronic signatures prescribe that the electronic act of signing allows to replace paper-based wet-ink signatures and that the electronic signature becomes legally recognised as equivalent to a handwritten one if:

- it allows the verification of a specific, i.e.uniquely identifiable, natural person[19] as the origin of the signed electronic document,
- it offers the detection of any subsequent modification, in other words integrity protection, and
- it signals the consent of the Signer to the contents of the signed electronic document.

It is this statement of legal equivalence found in Directive 1999/93/EC [175] and still in Regulation 910/2014 [182][20] that gives the signed electronic document a high probative value in the legal realm of the European Union. Hence, this thesis, backed by thorough analysis, will elaborate that the **detection of any subsequent modification** of a signed document is required by the legal framework. The legal background and the analysis to extract the requirements, as well as a suitable definition of integrity, are in Chapters 4 to 7. Together those chapters form Part I.

The 2nd goal is to be compliant with data protection requirements and allow to keep trade secrets confidential by having subsequent modifications, that are authorized by the malleable signature scheme, preserve the privacy of the original with a cryptographically sufficient strength. The thesis considers the following aspect of confidentially to be of key importance: Data protection for persons and trade secret protection for companies. Both aim to reduce the amount of information that is exchanged in a message to the minimum needed. If the data contains more information, the information is minimised by removing data that contains this information, such that the information remaining in the data afterwards serves the purpose and does not leak more information than necessary. This results in modified data. The cryptographic property termed privacy in the literature of RSS and SSS captures these requirements. Based on *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk*, who have formalised the privacy property [64], this thesis describes *standard* privacy as follows:

[17] A 'rant' by *Ellison* found on the Internet offers the following nice description: "**Cryptographic Non-repudiation** is provably achieved by all practical public-key cryptosystems. It is defined as being able to prove that if you have a digital signature that verifies with public key K [pk], then you know that the associated private key [sk] was used to make that signature. This definition says nothing about any human who might or might not have instigated the creation of that digital signature." [163].

[18] Henceforth, the term *subsequent modification* will be used in this thesis to denote that the action of modification was performed after the protection mechanism has been applied.

[19] Regulation 910/2014 introduced the concept of a "seal" [Section 4,Regulation 910/2014] where the signatory is a legal instead of a natural person, but to represent a legal person "a qualified electronic signature from the authorized representative of the legal person should be equally acceptable" [recital 58, Regulation 910/2014].

[20] Regulation 910/2014 and Directive 1999/93/EC are considered the two most influential European legal texts regulating electronic signatures within the EU; Regulation 910/2014 superseded Directive 1999/93/EC in mid 2016.

Definition 155 : Standard Privacy

> *A* **private** *scheme prevents* Verifiers *from recovering any information (esp. the original value) about block(s) of m, which are no longer in the sanitized block(s) of m', through a valid signature σ' over m'.*
>
> *Note: Information leakage through the semantic content of the modified message m' itself, which is given to the adversary, is out of scope.*

Hence, the second goal of this thesis is to provide cryptographically sound confidentially guarantees to fulfil the requirements found in legal texts on data protection, e.g., Directive 95/46/EC [174] and Regulation 45/2001 [176]. This thesis assumes that the protection offered by mechanisms that sufficiently remove personal data in the context of data protection are sufficient to protect organisational information as well, e.g. trade secrets. The legal background, the analysis and the extracted requirements, together with a definition that captures the confidentiality obtained for the original by subsequent can be found in Part II.

1.2.3. Examples for authorized subsequent modifications to documents

The following two examples are based on paper-based documents generated, signed and modified during the processes commonly found in food supply chains. For the first and second example they originate from scenarios researched during the author's participation in the German/French project ReSCUeIT[21]. To easily locate **examples** in this thesis they have been marked by a vertical line on the side of the text.

EXAMPLE

In a supply chain, the paper documents, which are created alongside the chain of physical goods exchanged, are subject to authorized changes or additions during their lifetime. Unchanged information is assumed to be still credible and gets accredited to the original author. Authenticity of documents plays a great role, as accountability in the food supply chain is key to meet the legal obligation of ensuring the traceability of food[22]. This means that the documentation must provide accountability for all food business operators[23] from whom food was received and to whom food was delivered. When those documents and associated processes are transformed into the digital realm, the paper documents become electronic documents and the process still demands the generation of evidence. This is highlighted in example one.

Removing information to keep it confidential and to hide trade secrets as well as personal information from prying eyes also becomes necessary during the supply chain: To settle disputes or resolve cases of food contamination, a supply chain partner needs to share such documents with a supply chain partner or a third party. This is highlighted with example two.

Finally, example three motivates that an electronic document must not only be electronically signed, but in specific application domains with a method that fulfils certain legal requirements such that it gives high probative value.

Example 1: When paper documents undergo subsequent modifications by authorized parties, they retain a high legal probative value. Fig. 3 shows the scan of a paper-based supply chain document. The document is the waybill[24] documenting that the bottles are transported by the freight forwarder from the bottle producer to the liqueur bottler. Namely the waybill documents type and amount of the goods. In this example the goods that this document refers to are glass bottles used to hold cherry liqueur for later human consumption. Hence, the bottles are part of the food supply chain. In

EXAMPLE

[21] German project title 'IT-Plattform für die lückenlose Sicherung von Lebensmittelwarenketten' `http://www.bbk.bund.de/DE/Service/Fachinformationsstelle/Informationsangebote/Forschungsberichte/ForschungsprogrammSicherheitsforschung/Sicherung_von_Warenketten/RescueIt/RescueIT_node.html` [last accessed: Jan. 2016].

[22] Art. 18 Para. 1 of Regulation (EC) 178/2002.

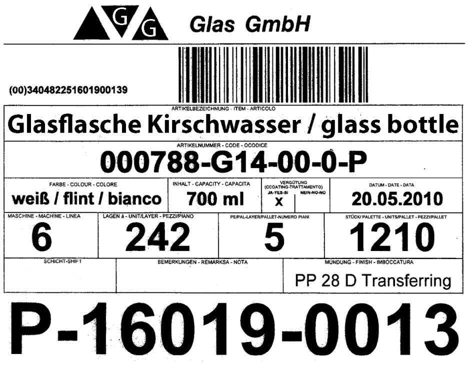
[23] A food business operator is "the natural or legal persons responsible for ensuring that the requirements of food law are met within the food business under their control" [Art. 3 of Regulation (EC) No 178/2002].

[24] Waybill is also known as *bill of consignment* or 'Frachtbrief' in German.

paper form the waybill documents are generated as three original paper copies.[25] One original for each of the three involved parties: sender, freight forwarder and recipient. The waybill usually comprises as a predefined section that acts as the proof of delivery (POD)[26].

The waybill must be signed by the sender[27] and the sender can demand that also the freight forwarder signs it. The waybill is evidence that the freight forwarder took charge of the goods in unharmed, undamaged condition[28]. The freight forwarder, on loading the stated amount of undamaged bottles, also signs the waybill.

Finally, when the goods are delivered, the dedicated recipient will sign the receipt of delivery to document the completed freight forwarding. In this case the liqueur bottler checks if the received bottles are complete in respect to their amount and if the goods are free of visual damages. If all is received in good condition the bottler signs, thereby creating a proof of delivery for the freight forwarder as well as for the glass bottle producer. Fig. 4 shows the paper document as an example of how such a waybill document that became a proof of delivery might look like.



Glas GmbH

(00)340482251601900139

ARTIKELBEZEICHNUNG - ITEM - ARTICOLO
Glasflasche Kirschwasser / glass bottle

ARTIKELNUMMER - CODE - OCODICE
000788-G14-00-0-P

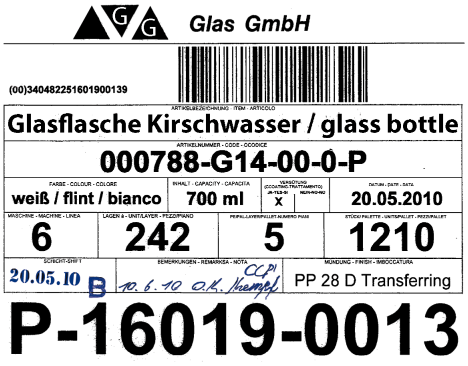
FARBE - COLOUR - COLORE: weiß / flint / bianco
INHALT - CAPACITY - CAPACITA: 700 ml
VERGÜTUNG (CCOATING-TRATTAMENTO) JA-YES-SI: X NEIN-NO-NO:
DATUM - DATE - DATA: 20.05.2010

MASCHINE - MACHINE - LINEA: 6
LAGEN à - UNIT/LAYER - PEZZI/PIANO: 242
PE/PAL-LAYER/PALLET-NUMERO PIANI: 5
STÜCK/ PALETTE - UNITS/PALLET - PEZZI/PALLET: 1210

SCHICHT-SHIFT:
BEMERKUNGEN - REMARKSA - NOTA:
MÜNDUNG - FINISH - IMBOCCATURA: PP 28 D Transferring

P-16019-0013

Figure 3. Original supply chain document: waybill issued by freight sender



Glas GmbH

(00)340482251601900139

ARTIKELBEZEICHNUNG - ITEM - ARTICOLO
Glasflasche Kirschwasser / glass bottle

ARTIKELNUMMER - CODE - OCODICE
000788-G14-00-0-P

FARBE - COLOUR - COLORE: weiß / flint / bianco
INHALT - CAPACITY - CAPACITA: 700 ml
VERGÜTUNG (CCOATING-TRATTAMENTO) JA-YES-SI: X NEIN-NO-NO:
DATUM - DATE - DATA: 20.05.2010

MASCHINE - MACHINE - LINEA: 6
LAGEN à - UNIT/LAYER - PEZZI/PIANO: 242
PE/PAL-LAYER/PALLET-NUMERO PIANI: 5
STÜCK/ PALETTE - UNITS/PALLET - PEZZI/PALLET: 1210

SCHICHT-SHIFT: 20.05.10 B
BEMERKUNGEN - REMARKSA - NOTA: 10.6.10 O.K. Kempf CCPl
MÜNDUNG - FINISH - IMBOCCATURA: PP 28 D Transferring

P-16019-0013

Figure 4. Subsequently modified document: recipients acknowledge receipt on the waybill creating the proof of delivery

The waybill is of economic importance as it can document the delivery and then serves as a receipt, when it is signed by the good's recipient.[29] By signing the waybill the signing parties declare that[30]:

(1) the physical good and its packaging showed no signs of damage and

(2) the number of pieces and

(3) their labels (signs, names, numbers) match the information in the waybill.

When a waybill is printed on paper and signed on paper it legally becomes a refutable presumption, meaning its correctness is initially assumed but can be rebutted. Thus, when migrating to an electronic waybill, the economic value must be preserved and thus the legal probative value for an electronic waybill shall be high. Additionally, the advantage of settling legal disputes quicker through a legal proceeding based on documentary evidence[31] is beneficial in the supply chain as it might allow especially smaller participants, which do not have the monetary foundation to fight lengthy court cases, to stay in business.

Within the supply chain processes, these paper-based documents are allowed to be updated after being signed by the bottle producer, e.g., corrected or amended. These updates shall only happen by dedicated parties. For example, the freight forwarder could subsequently modify the waybill's amount if during transport losses or damages occurred. And the recipient is supposed to modify the form fields that indicate how many goods it received undamaged. All these changes become evidence of what was

[25] "Der Frachtbrief wird in drei Originalausfertigungen ausgestellt, die vom Absender unterzeichnet werden." [§ 408 Abs. 2 HGB].

[26] Proof of delivery (POD) is also known as *receipt of delivery*; in German it is called 'Empfangsbekenntnis' or 'Quittung' which can be found in § 368 BGB.

[27] Following § 408 Abs. 2 HGB.

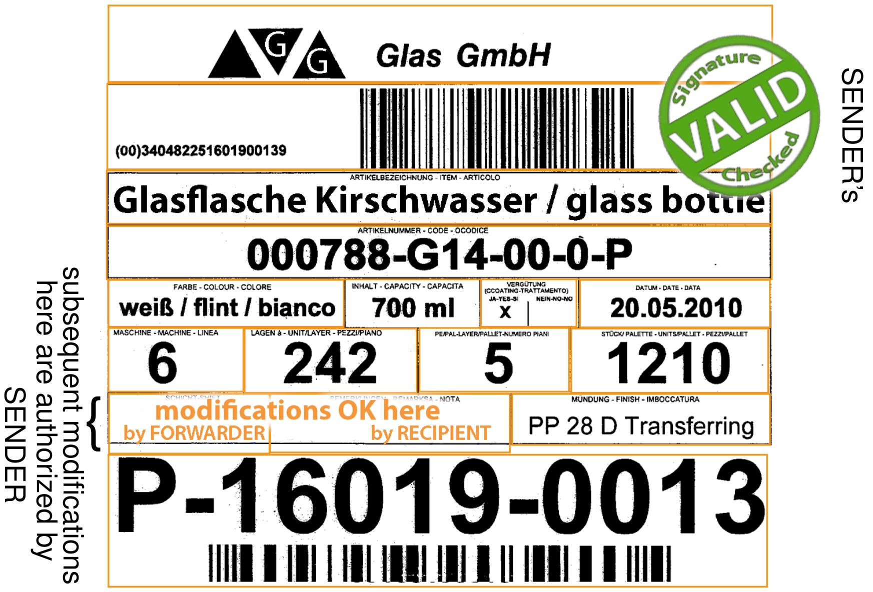
[28] § 409 HGB contains the waybill's evidentiary presumptions.

[29] following § 368 BGB

[30] following § 409 Paragraph 2 Sentence 1 HGB

[31] Following §§ 592 ff. ZPO n.F.; in German 'Urkundenprozess'.

actually delivered and allow the parties to settle disputes among them. On paper, the modifications are legally assumed to retain verifiability, allowing to trace the changes and identify the original issuer and the parties that changed that paper document (see Fig. 4). Hence, a sufficiently high probative value through technical mechanisms for integrity and authenticity of origin for all the modified and unchanged document blocks is important in an electronic counterpart of these supply chain documents.



Glas GmbH

(00)340482251601900139

ARTIKELBEZEICHNUNG - ITEM - ARTICOLO: Glasflasche Kirschwasser / glass bottle

ARTIKELNUMMER - CODE - OCODICE: 000788-G14-00-0-P

FARBE - COLOUR - COLORE	INHALT - CAPACITY - CAPACITA	VERGÜTUNG (CCOATING-TRATTAMENTO) JA-YES-SI	NEIN-NO-NO	DATUM - DATE - DATA
weiß / flint / bianco	700 ml	x		20.05.2010

MASCHINE - MACHINE - LINEA	LAGEN à - UNIT/LAYER - PEZZI/PIANO	PE/PAL-LAYER/PALLET-NUMERO PIANI	STÜCK/ PALETTE - UNITS/PALLET - PEZZI/PALLET
6	242	5	1210

SCHICHT-SHIFT | BEMERKUNGEN - REMARKSA - NOTA | MÜNDUNG - FINISH - IMBOCCATURA: PP 28 D Transferring

P-16019-0013

Figure 5. Document is split into several defined blocks; sender's signature authorizes only specific blocks to be subsequently modified by only the forwarder and the recipient

While the authorized modifications in those digitally signed documents are indeed desirable, they should always be limited to a third semi-trusted party and should occur in such a manner that the question of accountability can be addressed — as in this example. This requires to control the authorized modification. Namely, different blocks of the original document will be marked with various conditions, which have to be attained to in any subsequent modifications. When those changes can be controlled by the signer, this property is called signer control and it is later defined in Definition 117 on page 168 as follows:

Definition 117 : Signer Control

> *A malleable signature scheme offers* ***Signer control****, if the Signer, and only the Signer, can determine, which data is permitted to be subsequently altered. in which manner.*

Fig. 5 shows a potential breakdown of the supply chain document into blocks based on the form fields of the paper document. Moreover it indicates where subsequent modifications are authorized and by which party. In this example, two blocks are malleable as marked with 'modifications OK here' in Fig. 5. One is becoming the proof of delivery and must only be modified by the recipient of the goods.

Fig. 6 visualises that each sanitizing party from the above example becomes accountable for the modified document they have created jointly by their authorized modifications.

Speaking of accountability, it is legally important to distinguish a rightful, in other words "appropriate alteration" [347], from a non-authorized alteration. The applied signatures of the respective parties must still be verifiable if only authorized modifications were done. Note, in general an authorization contains an authorization for an actor to do an action, e.g., only the recipient (actor) shall modify rightmost of the two initially empty form fields (action) in Figures 5 to 7. Henceforth an authorization for a modification is comprised of information about the actors being authorized and the actions being authorized (see Sec. 3.7). Fig. 6 visualises that the document has only undergone authorized modifications, as all three signatures of all involved authorized parties still verify.

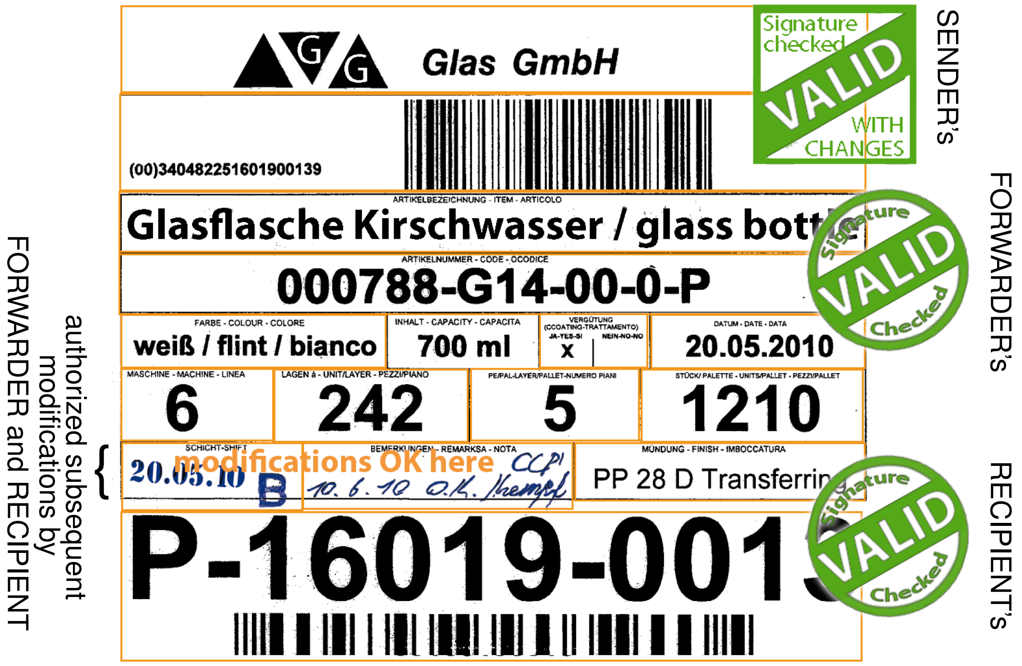

Glas GmbH
(00)340482251601900139
ARTIKELBEZEICHNUNG · ITEM · ARTICOLO
Glasflasche Kirschwasser / glass bott
ARTIKELNUMMER · CODE · OCODICE
000788-G14-00-0-P
FARBE · COLOUR · COLORE
weiß / flint / bianco
INHALT · CAPACITY · CAPACITA
700 ml
VERGÜTUNG (CCOATING-TRATTAMENTO)
JA-YES-SI
NEIN-NO-NO
X
DATUM · DATE · DATA
20.05.2010
MASCHINE · MACHINE · LINEA
6
LAGEN à · UNIT/LAYER · PEZZI/PIANO
242
PE/PAL-LAYER/PALLET-NUMERO PIANI
5
STÜCK/ PALETTE · UNITS/PALLET · PEZZI/PALLET
1210
SCHICHT-SHIFT
20.05.10
B
BEMERKUNGEN · REMARKSA · NOTA
CCP
10. 6. 10 O.K.
MUNDUNG · FINISH · IMBOCCATURA
PP 28 D Transferrin
P-16019-001
modifications OK here
Signature checked
VALID
WITH CHANGES
Signature VALID Checked
Signature VALID Checked
SENDER's
FORWARDER's
RECIPIENT's
authorized subsequent modifications by FORWARDER and RECIPIENT

Figure 6. Subsequently modified document contains verifiable signatures of sender, forwarder and receiver as only authorized modifications occurred

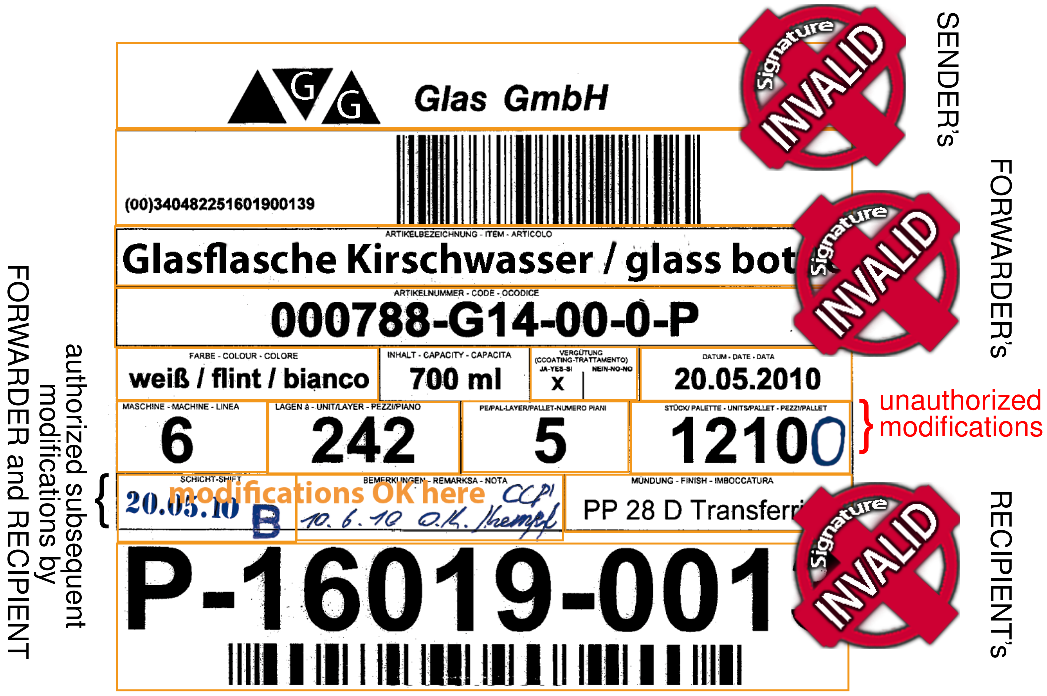
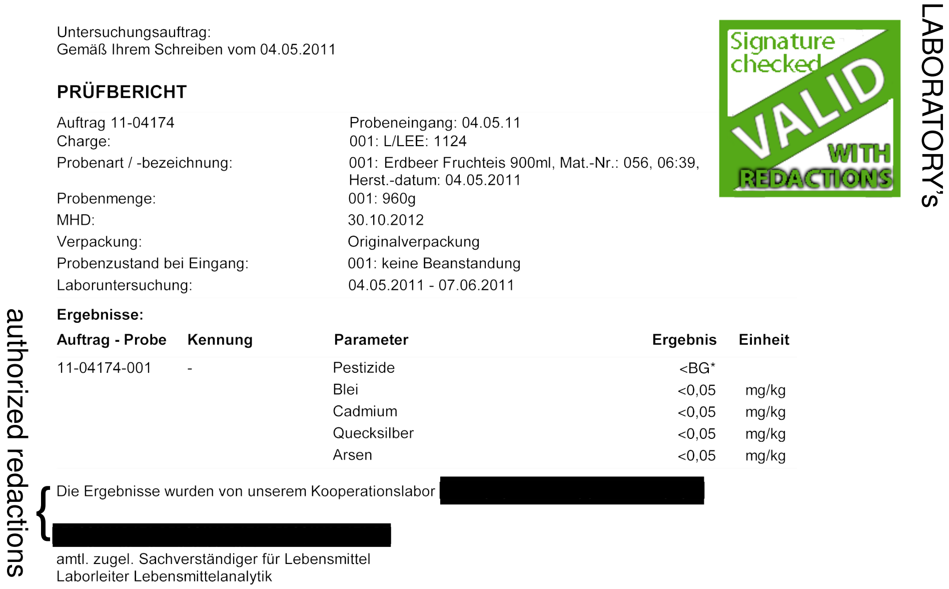

Glas GmbH
(00)340482251601900139
ARTIKELBEZEICHNUNG · ITEM · ARTICOLO
Glasflasche Kirschwasser / glass bot
ARTIKELNUMMER · CODE · OCODICE
000788-G14-00-0-P
FARBE · COLOUR · COLORE
weiß / flint / bianco
INHALT · CAPACITY · CAPACITA
700 ml
VERGÜTUNG (CCOATING-TRATTAMENTO)
JA-YES-SI
NEIN-NO-NO
X
DATUM · DATE · DATA
20.05.2010
MASCHINE · MACHINE · LINEA
6
LAGEN à · UNIT/LAYER · PEZZI/PIANO
242
PE/PAL-LAYER/PALLET-NUMERO PIANI
5
STÜCK/ PALETTE · UNITS/PALLET · PEZZI/PALLET
12100
SCHICHT-SHIFT
20.05.10
B
BEMERKUNGEN · REMARKSA · NOTA
CCP
10. 6. 10 O.K.
MUNDUNG · FINISH · IMBOCCATURA
PP 28 D Transferri
P-16019-001
modifications OK here
Signature INVALID
Signature INVALID
Signature INVALID
SENDER's
FORWARDER's
RECIPIENT's
unauthorized modifications
authorized subsequent modifications by FORWARDER and RECIPIENT

Figure 7. Unauthorized modifications, even when done by authorized parties, invalidate the signature(s)

Fig. 7 visualises that an unauthorized modification to the signed document results in a failed signature verification, e.g. here, an entry was changed in a part of the document which was not authorized to be subsequently modified.

EXAMPLE

Example 2: Information is subsequently removed from authentic documents to protect trade secrets or personal information with the goal to retain an increased probative value for the remaining information. Removing parts of the document to hide certain information also becomes necessary for documents from the supply chain. However, if the authenticity of the remaining information cannot be kept, then a removal will not take place and information is leaked. In this second example, the document shown in Fig. 8 is the laboratory report for an ice cream sample. Such a document is usually part of the documentary information that is kept by the ice cream producer during the production and after shipment. Consider the case that the food authority wants to investigate in a case of food contamination. For the sake of this example, assume that the same ice cream product is currently under investigation for being dangerously contaminated because children that ate it in a kindergarden suffered from a rash. At latest when the authorities are investigating the cause it becomes the duty of food processors to provide the necessary documentation. In most cases, even before the authorities ask for documents, all food processors in that supply chain want to establish evidence that allows to rebut the accusation that the cause for the rash was indeed the way the ice cream was produced, transported, stored and sold. Hence, the supply chain partners need to share such documents among each other and at some point with the food authority. Latest on request of the food authority, e.g., to determine information about the flow of goods that are involved in a food crisis, information like the waybill from the previous example must be provided[32].


Signature checked
VALID
WITH
REDACTIONS
LABORATORY's
authorized redactions

Untersuchungsauftrag:
Gemäß Ihrem Schreiben vom 04.05.2011

PRÜFBERICHT

Auftrag 11-04174	Probeneingang: 04.05.11
Charge:	001: L/LEE: 1124
Probenart / -bezeichnung:	001: Erdbeer Fruchteis 900ml, Mat.-Nr.: 056, 06:39, Herst.-datum: 04.05.2011
Probenmenge:	001: 960g
MHD:	30.10.2012
Verpackung:	Originalverpackung
Probenzustand bei Eingang:	001: keine Beanstandung
Laboruntersuchung:	04.05.2011 - 07.06.2011

Ergebnisse:

Auftrag - Probe	Kennung	Parameter	Ergebnis	Einheit
11-04174-001	-	Pestizide	<BG*	
		Blei	<0,05	mg/kg
		Cadmium	<0,05	mg/kg
		Quecksilber	<0,05	mg/kg
		Arsen	<0,05	mg/kg

Die Ergebnisse wurden von unserem Kooperationslabor [redacted]

[redacted]

amtl. zugel. Sachverständiger für Lebensmittel
Laborleiter Lebensmittelanalytik

Figure 8. Snippit from a food contamination report issued by an authorized testing laboratory; The name of the person signing the report was considered personal information and thus got removed (bottom black square); The name of the company that was subcontracted for some tests (in German called 'Kooperationslabor') was considered a trade secret and thus got removed (upper black square); Still being able to positively validate the laboratory's signature retains the probative value for the remaining information (among them are the test results for pesticides ('Pestizide'), lead ('Blei'), cadmium ('Cadmium'), quicksilver ('Quecksilber') and arsenic ('Arsen')).

Assume that the supermarket chain for which the ice cream was produced is now asking participants within the supply chain. Namely, the ice cream producer could be asked to show the mandated[33] laboratory tests. This information might be known to be forwarded to the public or the authorities if the results show evidence that the goods produced were not subject to any contamination. Thus it is important that the documentation shall contain the information about the laboratory tests done which indicate that the ice cream showed no signs of contamination at that point in the food supply chain. This is the kind of important evidence that the ice cream reseller is looking for to rebut that the ice cream was the reason for the children's rash. To help clearing the case, e.g., for ethical reasons or due to legal or business demands, the company participating in the supply chain in question is willing to supply this as necessary information asked for by the food authorities. Of course, as a necessary documentation obligation the integrity and authenticity of the information provided to the authorities shall be verifiably intact when verified by the authorities, e.g., the report shall be unadulterated and come from a laboratory that is legally allowed to do those tests — all this is adding necessary credibility to the document's contents.

[32] Art. 18 Para. 2 Sentence 2 and Para. 3 Sentence 2 of Regulation (EC) 178/2002 [181].

[33] Those documents might not be mandated by legal texts, but by tight contractual obligations. At least it was identified as being demanded in the German food supply chains investigated during the ReSCUeIT project.

However, in this example the original report[34] contained additional information that is unnecessary to be transmitted and the party having the original report thus wants to redact this unnecessary information before sharing it. In this example, the name of the natural person that signed the report as her or his duty when working for the laboratory, as well as information about another sub-contracted laboratory[35] shall be removed. The latter could be regarded as a trade secret belonging to the laboratory and thus need removal. The former could be removed to protect the employee's personal privacy.[36]

Both redactions might be required to be revealed later if requested by authorities if the laboratory itself or the person doing the tests would be under investigation. This example concludes with the assumption that if the remaining non-redacted information of the laboratory report has enough probative value then the food authority is convinced; and hence the ice cream producer is eliminated as the cause for the contamination in this supply chain.

This second example motivates that retaining a verifiable signature on the remaining unchanged information allows to establish trust in unchanged remaining partial data if one is able to verify it with just the original trust anchor, e.g., the laboratory's electronic signature remains verifiable. It further shows that limiting the authorized modifications to only subsequent redactions — far more restrictive than allowing any modification as with the sanitizations authorized in example one — gives room for many applications that require to re-tailor the information contained in authentic documents before releasing them to third parties without completely needing to sacrifice the authenticity.

EXAMPLE

Example 3: Legal requirements in specific application domains demand electronic documents to be electronically signed with a method that gives high probative value. The third example motivates that electronic documents must not only be electronically signed, but with a method that fulfils certain legal requirements. Fig. 9 depicts a screenshot of an electronic document containing a credit note which was issued after returning goods for which an invoice stating value added tax (VAT) was issued and paid. It is electronically signed and the signature can be validated.

Indeed, electronic documents related to financial tax are an example for documents that legally have been put under more stringent technical requirements. For example, "[t]he authenticity of the origin, the integrity of the content and the legibility of an invoice, whether on paper or in electronic form, shall be ensured from the point in time of issue until the end of the period for storage of the invoice." [Art. 233 para. 1 Directive 2010/45/EU]. This can be achieved by "[...] an advanced electronic signature within the meaning of point (2) of Article 2 of Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures (5), based on a qualified certificate and created by a secure signature creation device, within the meaning of points (6) and (10) of Article 2 of Directive 1999/93/EC" [Art. 233 para. 2 a Directive 2010/45/EU]. In Germany, until 2011, § 14 UStG [90][37] directly required a qualified electronic signature on electronic invoices in order to claim for input tax deduction[38].

During the course of this thesis several court cases where found that re-iterated the need for legally qualified electronic signatures to be be generated: The German Federal Fiscal Court (BFH) ruled that in Hamburg the court must accept the filing of an action[39] using electronic mail (email) only if the email is signed using a qualified electronic signature that is in accordance with German signature law § 2 Nr. 3 SigG[40]. Another court ruling in this area is from the higher regional court (OLG) of Munich; it ruled that

[34] Fig. 8 is taken from a real supply chain transaction researched as part of the ReSCUeIT project, but with minor changes to preserve space.

[35] Translated from the German 'Kooperationslabor'.

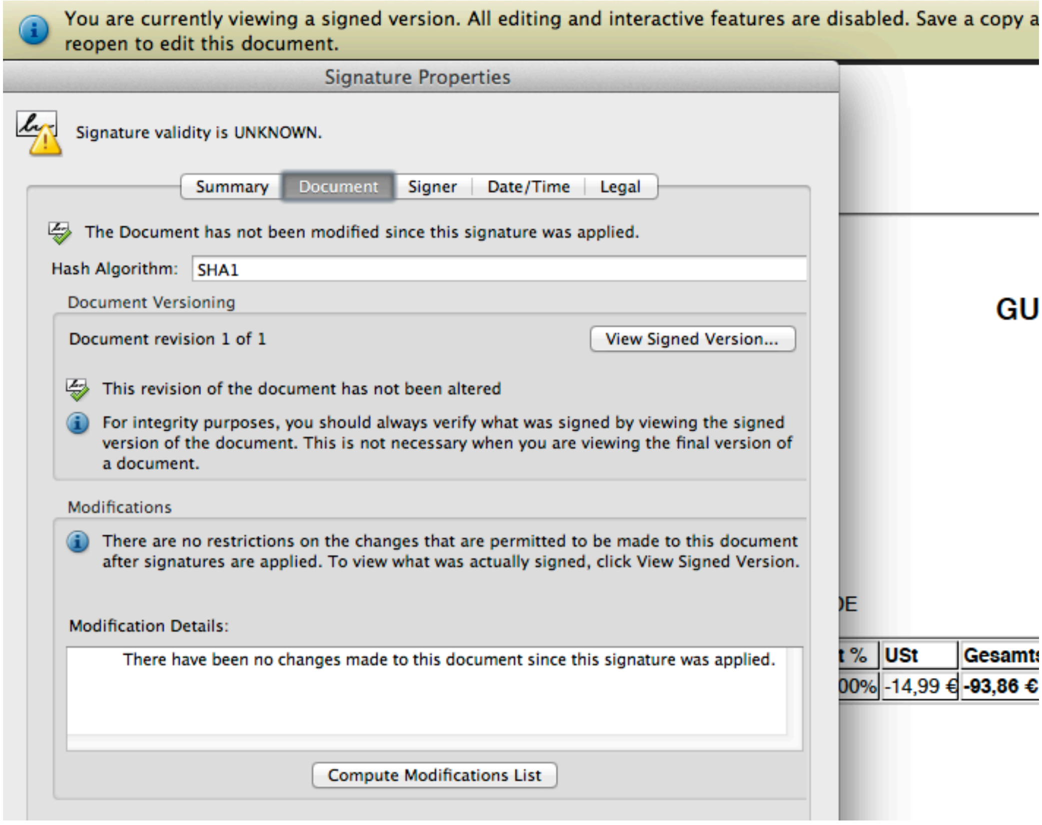
[36] German legal texts state that redaction can be used to remove data: See the court order from the superior Hessian administrative court (Hessian VGH) in Germany from 02.08.2012 - Az. 27 F 96/11: "Die Verweigerung der Nennung schutzwürdiger Daten Dritter in folgenden Unterlagen ist rechtmäßig. Sie ist durch Schwärzung zu gewährleisten [...]" [228]. See further the statements made by the German Federal Administrative Court (BVerwG) in the ruling from 23.06.2011 - Az. 20 F 21/10, Randnummer 22. Herein the court also positively described the use of partial redactions to keep secrets stating "Geheimhaltungsbedarf ist jedoch grundsätzlich zu bejahen [...] Die Passage enthält Daten, die Rückschlüsse über die wirtschaftliche Leistungsfähigkeit des genannten Instituts erlauben." [101]. Furthermore the BVerwG argued that names of governmental personnel in a report can receive confidentiality protection and hence may be removed. The remaining partially redacted document shall still be acceptable: "[...] Soweit der Bericht mit den Namen behördlicher Mitarbeiter personenbezogene Daten enthält, greift ebenfalls ein 'wesensmässiger' Geheimhaltungsgrund, dem durch teilweise Schwärzung Rechnung getragen werden kann. [...]" [101].

[37] The old version of the German tax laws [90] from 2005 ceased to have effect in 2011.

[38] Tax deduction translates to 'Vorsteuerabzug' in German.

[39] Translation of 'Klageerhebung' according to Langenscheidt Recht Englisch [70].

[40] BFH, ruling 26.07.2011 VII R 30/10 [77].


You are currently viewing a signed version. All editing and interactive features are disabled. Save a copy a reopen to edit this document.

Signature Properties

Signature validity is UNKNOWN.

The Document has not been modified since this signature was applied.

Hash Algorithm: SHA1

Document Versioning

Document revision 1 of 1

This revision of the document has not been altered

For integrity purposes, you should always verify what was signed by viewing the signed version of the document. This is not necessary when you are viewing the final version of a document.

Modifications

There are no restrictions on the changes that are permitted to be made to this document after signatures are applied. To view what was actually signed, click View Signed Version.

Modification Details:

There have been no changes made to this document since this signature was applied.

Figure 9. Screenshot showing Adobe Acrobat Pro Version 9 on Mac OSX with the output when validating a signed PDF with a credit note from Amazon EU S.á r.l.

a contract to take up a loan must be either paper-based carrying a handwritten signature or in electronic form with a qualified electronic signature following § 126 BGB. Another digital and cryptographically much weaker form of a signature — in this case the customer signed with a pen on an electronic tablet — was not sufficient to make the electronic document become a legally binding contract[41]. Altogether, this third and final example motivates that being able to generate electronic signatures such that they fulfil high legal requirements is important for legal accuracy in many application domains. Hence, if a cryptographic scheme is not able to achieve the legal requirements then this scheme would simply not be usable in practice in that domain.

1.3. Research question

As motivated, the subsequent authorization of modifications made possible by malleable signature schemes (MSS) offer additional functionality which is beneficial in several application domains. MSS cryptographically enhance digital signatures. However useful MSS might be technically, the reasons for applying electronic signatures are most often to establish legal accountability. In general, generating an electronic signature or verifying one means additional overhead, i.e., additional compute power, additional storage at rest and in transmission. Hence, when applied, they shall increase the probative value for the protected data to offer a benefit from a business perspective. In some applications, as example three highlighted, the reason for applying electronic signatures are legal obligations. Legal texts — i.e., national laws, European directives and regulations — set forth what technical mechanisms must achieve to obtain a high probative value for documents in electronic form (digital data). Additionally to

[41] OLG München, ruling 04.06.2012 19 U 771/12 states "[...] ergibt sich ausdrücklich, dass Verbraucherdarlehensverträge [...] nicht nur in Schriftform mit Unterschrift, sondern auch durch qualifizierte elektronische Signatur (§ 126 a BGB) abgeschlossen werden können sollten [...]" [363, Abs. 20 (5)] und "[...] andere elektronische Dokumente, zu denen auch ein eigenhändig unterschriebenes Dokument auf einem elektronischen Schreibtablett zählt, zur Wahrung der Form nicht ausreichen sollten" [363, Abs. 20 (6)].

meeting the legal requirements for an electronic signature, the malleable signature must protect the confidentiality from a sophisticated adversary. The authorized subsequent modification shall offer strong cryptographic confidentiality protection for the original data that got modified. Hence, the underlying cryptographic mechanism must be mathematically proven to withstand such attacks.

The research question underlying this thesis is as follows:

> Can a malleable signature scheme be private to be compliant with EU data protection regulation **and at the same time** fulfil the integrity protection legally required in the EU to achieve a high probative value for the data signed?

As a consequence, the malleable signature schemes of interest must generate added value by achieving both of the following:

- Integrity preservation and accountability guarantees must be allowed to be validated — technically unambiguously — by an independent party, e.g., by a judge in court.
- Confidentiality protection must be achieved by allowing subsequent modifications of (even integrity-protected) data such that a trade secret or personal information contained in the original data is no longer reconstructable from the modified data.

1.4. Scope

The properties of integrity and authenticity are defined as theoretic properties and thus are neutral towards the technological mechanism to achieve them. The legal scope of this thesis covers the European Union's laws and regulations. For the technical mechanism, this thesis concentrates on specialised digital signature schemes that allow subsequent authorized modifications. This scope was selected for the following two reasons:

- Digital signatures are already used in practice to achieve the goals of integrity and authenticity; and technical standards exist (see also Analysis Result 13).
- There is EU legislation dedicated to electronic signatures, e.g., Regulation 910/2014 and formerly Directive 1999/93/EC. And there are legal cases, e.g., [81], that discuss technical mechanisms' legal interpretation regarding the probative value for the signed data (see also Analysis Result 3).

1.4.1. Technical scope

This thesis concentrates on so-called malleable signature schemes (MSS) as special detective integrity protection mechanisms without message recovery[42]. Herein, the thesis focuses on redactable (RSS) and sanitizable signature schemes (SSS) in the single sanitizer framework[43]. The *malleability* is important for solving the digital document sanitization problem.

The thesis is treating RSS and SSS as two different malleable signature schemes. While they seem to be very much alike, they are different classes and must be treated separately as shown in Chapter 15.

The thesis is focussed on RSS and SSS schemes with a single entity in the role of the Sanitizer. The restriction to a single sanitizer allows for easier cryptographic proofs and increases readability. This is mostly due to the fact that only one sanitizer and hence only one sanitizer key pair exists. For selected topics this thesis briefly mentions the multi-sanitizer framework, e.g., Sec. 7.4.3. Also legally, the multi-sanitizer framework does not introduce too many new obstacles, and is thus identified as future work (see Sec. 19.3).

This thesis requires cryptographic schemes to offer provable security for the cryptographic algorithms of the RSS and SSS. This means that the security objective needs to be formally defined: This thesis uses game-based definitions (see Sec. 2.3.1 on page 22). A formal proof is provided that reduces the security of the formally defined protocol to the security of a primitive used as an underlying building block which has a known well-defined strength (see Sec. 2.3.1 on page 22).

[42] Integrity services which include recovery mechanisms are not excluded, but this thesis will not look at them specifically.

[43] The multi-sanitizer framework was introduced by *Canard et al.* [112].

The focus is also on *detective* integrity protection mechanisms. This allows for the “detection of integrity compromise” [340], as opposed to mechanisms concerned with “prevention” [340] of integrity breaches[44]. Legally, it could be sufficient to establish evidence that subsequent modifications could not have occurred or that only authorized subsequent modifications have occurred due to technical protection working as intended. If this could be argued, then a sufficiently high probative value can possibly be given to documents treated within those protected systems[45]. Many of the practically used cryptographically based systems which additionally have been directly or indirectly subject to legal court cases, been mentioned in legal texts or been discussed in jurisprudence, are detective mechanisms based on cryptographic schemes built on asymmetric cryptographic primitives: Digital signatures[46]. Digital signatures technically provide evidence[47] that a certain secret key was involved in creating the signature which can be presented to a third party. This focus on detective mechanisms is further helpful as within the class of digital signature schemes there are already technical implementations which are practically used and legally recognised, e.g., RSASSA-PSS from PKCS-v2.2 [422].

1.4.2. Legal scope

The European Union’s (EU) legislation recently updated legal texts concerned with the two concepts that the research question is touching: electronic signature and data protection. Fig. 10 shows the position of the most relevant legal texts analysed for this thesis in the hierarchy of norms within the EU.

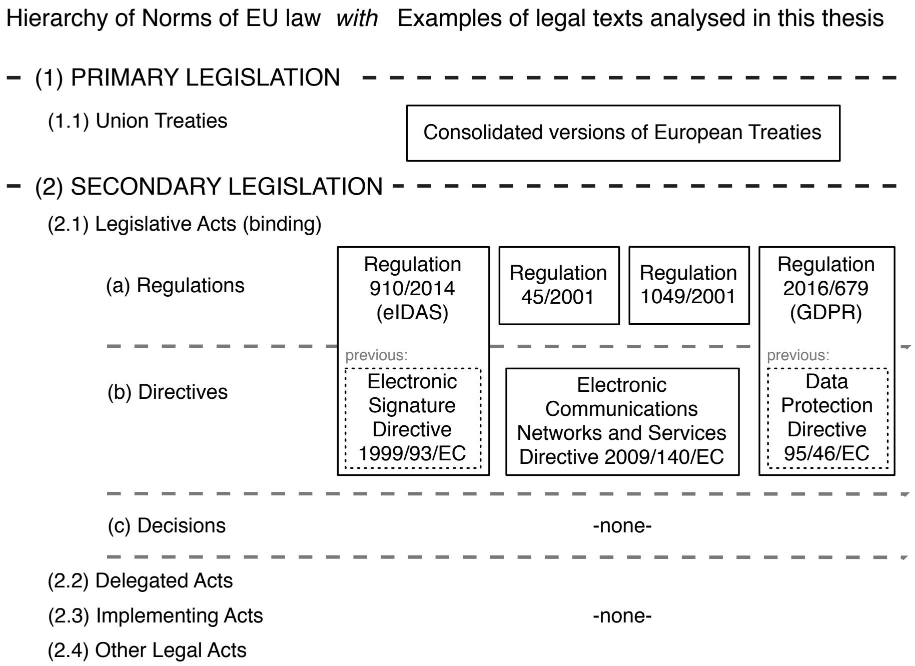

Hierarchy of Norms of EU law *with* Examples of legal texts analysed in this thesis
(1) PRIMARY LEGISLATION
(1.1) Union Treaties
Consolidated versions of European Treaties
(2) SECONDARY LEGISLATION
(2.1) Legislative Acts (binding)
(a) Regulations
Regulation 910/2014 (eIDAS)
Regulation 45/2001
Regulation 1049/2001
Regulation 2016/679 (GDPR)
(b) Directives
previous:
Electronic Signature Directive 1999/93/EC
Electronic Communications Networks and Services Directive 2009/140/EC
previous:
Data Protection Directive 95/46/EC
(c) Decisions
-none-
(2.2) Delegated Acts
(2.3) Implementing Acts
-none-
(2.4) Other Legal Acts

Figure 10. European Hierarchy of Norms and the legal texts discussed in Chapter 4 and in Chapter 8

[44] Following *Gollmann*, in general three classes of protection mechanisms for the security protection of assets can be distinguished: prevention, detection and reaction [214]. However, even if preventative integrity protection mechanisms are employed, e.g., access control, detective mechanisms are needed to provide sufficient evidence that the preventive integrity protection was working. *Gollmann* also notes that “there is not always a direct trade-off between prevention and detection” [214].

[45] Technically, access control can prohibit unauthorized access to functionality that allows to modify information in a system. Legally, access control in financial auditing systems is mandated by the German principles regarding the proper keeping and preservation of books, records and documents in electronic format and regarding data access (GoBD) [83].

[46] See also ISO 10181-6 stating “8.1.2 Integrity provision through Digital Signatures” [340].

[47] See also the discussion on ISO 13881 in Definition 91.

For signature legislation the focus is on the latest updated EU legal texts in the form of Regulation 910/2014; this thesis still uses the previous EU Directive 1999/93/EC for historical purposes and because older analysis was based on it[48]. Both address the entire EU; both are written in technology-neutral language [175, 182]. Additionally, this thesis will also briefly look at definitions in US regulations and within the EU treaty. Following the EU Regulation a so-called qualified electronic signature must be created to obtain a high probative value. In more detail, evidence laws in each member state govern how the probative value is assigned to electronically signed documents [172, 280, 284][49].

As a specific instantiation of member state law, this thesis will use legal texts of Germany (e.g., ZPO n.F., BGB, SigG[50], VDG, ZPO a.F.[51]) to analyse the assignment of probative value, as well as court rulings regarding electronic signatures in Germany. The former EU Directive 1999/93/EC still needed to be transposed into member state law (e.g., SigG), but the current Regulation 910/2014 becomes directly applicable and the German SigG and SigVO have ceased to have effect[52] end of July 2017[53]. This narrow field of vision was indispensable in order to allow a precise analysis. The ZPO and other legal texts were analysed using grammatical interpretation to check for the treatment of authorized changes[54]. The electronic signature is legally recognised as equivalent to a handwritten wet-ink signature if it is generated using an existing, cryptographically strong digital signature scheme, and if it is generated on a qualified signature creation device, such as a secure smart card, and if the public key is certified by an accredited certification authority, such as those listed by the European Union in [170]. The previous holds only unless the law explicitly demands a paper-based document. The recognised technical implementation for this is a classic asymmetric-key-based digital signature.

For data protection legislation this thesis started with a focus on the Directive 95/46/EC. However, as this will get superseded, whenever possible, it takes into account the recently released Regulation 2016/679 [184], better known as General Data Protection Regulation (GDPR). The GDPR was adopted on 27*th* of April 2016 and will become directly applicable[55] on 25*th* of May 2018 for all EU Member States. MSS are a privacy-enhancing technology (PET) as they help to minimise information disclosure by allowing for subsequent modifications. A positive answer to the research question would allow to substitute standard electronic signatures by an MSS. Thus, RSS and SSS with a high probative value would be a valuable tool to exercise the data minimisation principle also for signed data with a need for an increased probative value. The principle of data minimisation is prominently mentioned in EU data protection legislation [174, 176, 184][56].

1.4.3. Out of scope

The following four problems in the domain of signatures are not in the scope of this thesis: Linking cryptographic keys to legal entities including key management, proving the signatories intention to sign, the presentation problem, the secure implementation, deployment and use of the cryptographic algorithms and finally the identification of what information has to be redacted. The thesis will point to existing solutions where appropriate or to mechanisms considered *state of the practice*[57] that meet the current legal requirements. However, those mechanisms are out of scope and must be additionally deployed to make MSS meet the requirements for a PET or give an increased probative value.

[48] Regulation 910/2014 (eIDAS) has replaced Directive 1999/93/EC in mid 2016.

[49] In 2000 the European Commission noted "[...] that evidence laws can present barriers [...]" [172, p. 104].

[50] This thesis notes that this German legal text has just recently (29*th* of July 2017 [98]) ceased to have effect, but it is analysed as among other older discussions in the legal debate are based on it.

[51] This thesis notes that this German legal text has just recently (18*th* of July 2017) been updated [99], but an earlier version [96] got analysed as well because older discussions in the legal debate are based on the former version.

[52] The German legal term 'außer Kraft treten' was translated following *Dietl and Lorenz* [152].

[53] Namely, on 29*th* of July 2017 [98] SigG and SigVO ceased to have effect in Germany.

[54] *Jungermann* [277] wrote a more general legal analysis of the probative value induced by the ZPO in 2002.

[55] Unlike a directive, a regulation does not require any legislative changes by EU Member States to become applicable.

[56] Directive 95/46/EC [174] and GDPR [184] directly use the term data minimisation; Regulation 45/2001 [176] refers to it as "not excessive in relation to the purposes for which they are collected and/or further processed;" [Art. 4, 1(d) Regulation 45/2001].

[57] This expression is used to refer to what is currently used in practice, in comparison to the expression *state of the art* which is used to refer to what is the latest in scientific research.

Establishing the link between legal person and cryptographic key and key distribution is out of scope. The technical and organisational means for linking a natural person or legal entity to cryptographic keys are out of scope. This means that public key infrastructures (PKI) or other means of public key management are in place and are assumed to be working. This is stated as Security assumption 2 (see Sec. 3.8.3).
Note that the assumption that the legal signatory was affixing his or her signature to a document based on a valid electronic signature is debatable [330][58]. The current PKI has weaknesses, as pointed out for example by *Gutmann* [220][59], or by *Lopez, Oppliger, and Pernul* [320][60], or by *Ellison and Schneier* [162]. Fixing this is a problem of its own, e.g., see *Brands* [63]. However, as it is current legal practice [170][61] and business practice [8, 45] to use an infrastructure of accredited certificate authorities, MSS mechanisms that can work upon the existing infrastructure offer a lower adoption barrier. Hence, it is one objective of this thesis to cryptographically instantiate RSS and SSS such that their keys can be integrated with the existing PKI infrastructure.

Establishing the technical proof of the intentional signing of the document is out of scope. Foremost, it is out of scope as it cannot be achieved by cryptographic methods alone, which are the focus of this thesis. Achieving it requires the inclusion of organisational methods and consider human computer interaction (HCI), henceforth called *additional measures*. Hence, this thesis assumes that existing additional measures, which are solutions for classical DSS, get re-used for MSS. Still, establishing a proof of intention is one of the functions that hand-written signatures fulfil and thus the electronic signatures have to provide as well. However, it is legally accepted that current legally recognised digital signature schemes can achieve this by adding flanking technical and organisational measures. The most notable of those additional measures is the qualified signature-creation device (QSCD)[62] that protects against the unauthorized and unintended usage of the secret signature key. Usually, an additional password or PIN[63], only known to the signatory[64], has to be provided to the QSCD for each signature it generates. It is assumed that needed additional measures can be built in the same way for the MSS as they are built for DSS. MSS have one peculiarity: The additional measures must further ensure that the signatory's intention to authorize certain subsequent modifications by an MSS is captured at the time of signature generation.

Solving the presentation problem is out of scope. Even a PIN and a qualified signature-creation device (QSCD), which protect against the unauthorized usage of the involved cryptographic secrets by others than the signatory, cannot by themselves establish that "what you see is what you sign" [306] or short "WYSIWYS" [306]. WYSIWYS is also known as the presentation problem and is surely a problem [330, 399, 432], but out of scope for this thesis. However, a solution [276, 315, 364, 369] that solves this problem for an implementation of a legally recognised classical digital signature scheme (CDSS) is assumed to work for MSS for the following reasons: MSS can use the same trusted hardware as a CDSS, e.g., for secret key storage[65]. MSS can use the same kind of software implementations

[58] *Mason* even goes as far as stating that non-repudiation towards the natural person is impossible; only "[i]deally, there ought to be a technical method in place that prevents the person appending the signature to the document from claiming later that they did not sign it. This is virtually impossible to achieve in the electronic environment. Care must be taken to distinguish between the degree of probability that a system can be designed to prevent a person from making such a claim, and any suggestion of a presumption that purports to bind the user to a signature that is verified." [330].

[59] This was pointed out in [220] but can also be found in several other of his writings listed on *Peter Gutman*'s homepage `https://www.cs.auckland.ac.nz/~pgut001/` [last accessed: Jan. 2016].

[60] While their work from 2005 is titled "Why have public key infrastructures failed so far?" it still contained the remark that "Public key cryptography in general, and digital signatures and public key-based key establishment procedures are simply too valuable than not to be used in practice." [320].

[61] Information about European Union's approach of Trusted Lists of Certification Service Providers (TLSs) to bootstrap the hierarchical trust into certification authorities can be found online: `https://ec.europa.eu/digital-agenda/en/eu-trusted-lists-certification-service-providers`.

[62] This is also generally known as a hardware security module (see Definition 29).

[63] PIN stands for 'personal identification number'.

[64] Also out of scope is the problem that even if a password or PIN are used, those can be handed to other humans by the signatory, which might be considered intentional delegation as well, but this is not considered in this thesis and is not always legally tolerated. As an example: A German lawyer was not allowed to delegate his SSCD (as it was called in SigG) by giving the PIN to his secretary following a ruling from the BGH from 21.12.2010, Az.: VI ZB 28/10 [80].

[65] See especially Sec. 13.10 and Sec. 14.15 of this thesis for a proof of concept to demonstrate the usability of commercially available secure smart cards for signature generation with an MSS.

as a CDSS for the communication with the user. Additionally to the presentation problem in general, the MSS implementation must clearly communicate to ensure the signatory understands the process and consents to all the authorizations for subsequent modifications that are endorsed by generating the signature with an MSS instead of a CDSS.

The secure implementation, deployment and use of the cryptographic algorithm is out of scope. It is not part of this thesis to discuss a secure implementation of the algorithms nor their secure deployment, integration or use. This is just assumed to not introduce any weaknesses. This thesis offers formal security proofs which establish that the proposed schemes are provably secure (see Sec. 2.3.1). Note, this is a required baseline, because a cryptographically broken scheme is not generally fixed by implementing it. This does not mean exclude that an algorithm's flaw can be fixed by a clever implementation but this fixed implementation then implements at least a more tightly specified algorithm or even a different one. The thesis discusses this limitation in Sec. 2.3.3.

Identifying which information is to be protected by sanitization or redaction is out of scope. Lastly, it is out of the scope of this thesis to identify what information from which documents is regarded as personal data or as trade-secrets. This thesis escalates this important decision to the application domain and eventually finally to a human expert.[66] The RSS and SSS presented in this thesis serve as a starting point. They become the cryptographically sound mechanisms to build into software that enhances services and processes with the added functionality offered (see Sec. 18.1 for an application example.).

1.5. Methodology and interpretation of legal texts

The flow chart in Fig. 11 depicts the six steps, with some feedback loops, that were followed when conducting the research for this thesis. Step 1 captures with sufficient technical terminology the technical requirements. This requires technical precise terminology and as this was missing the first feedback-cycle occurs with an iteration of steps 2.1 and 2.2. The resulting terminology can be used to formulate each legal or application requirement found. The terminology is further used in Step 3.1 to harmonise the properties of existing schemes. This allows in step 3.2 to compare them with the requirements. Thereafter, step 3.2 identifies gaps in security and functionality that existing sanitizable (SSS) and redactable signature schemes (RSS) do not yet cover. To close those gaps the second cycle over the steps 4, 5 and 6 is iterated. This results in an advancement of the State of the Art in SSS and RSS where necessary to provably fulfil the requirements. A sub-feedback-cycle surrounds step 4, which subsequently defines new cryptographic properties. As a result, this methodology yields:

- technical requirements to comply with legal obligations,
- a refined terminology of integrity,
- harmonised SSS and RSS properties,
- new cryptographic properties for SSS and RSS, and
- new cryptographically proven SSS and RSS constructions that achieve the formally defined properties to fulfil the requirements.

The process to transform the meaning of legal concepts into technical requirements is as follows: Starting point were legal texts that defined or described the legal goals of "Integrity" and "Authenticity" in general (see Sec. 4.3) and with the specific context of "Electronic Documents" (see Chapter 7). To provide readers with enough insight into the legal texts used for extracting definitions, this thesis will quote the according parts of legal texts mostly in full, waiving the need for readers to look up the relevant passages in directives or laws on their own. Then the thesis will **first** give a *grammatical interpretation*: Based on the literal words one extracts the meaning of the legal text[67].

[66] *Chakaravarthy et al.* [115] present a technical solution to select natural language message's contents for redaction. *Chow et al.* [124] present a tool to support the user at the task of sanitization and redaction of text. They "leverage data mining and linguistic parsing to automate the privacy-risk analysis and make suggestions as to how to revise the document." [124]. Both works show that tools could support the work of well-educated humans in carrying out sanitizations or redaction properly. Also consider the work of *Sweeney* [462] and related works regarding k-anonymity.

[67] The word "grammatical" seems to be commonly used for this type of interpretation, cmp. *Black* [52] or *Lazaratos* [309].

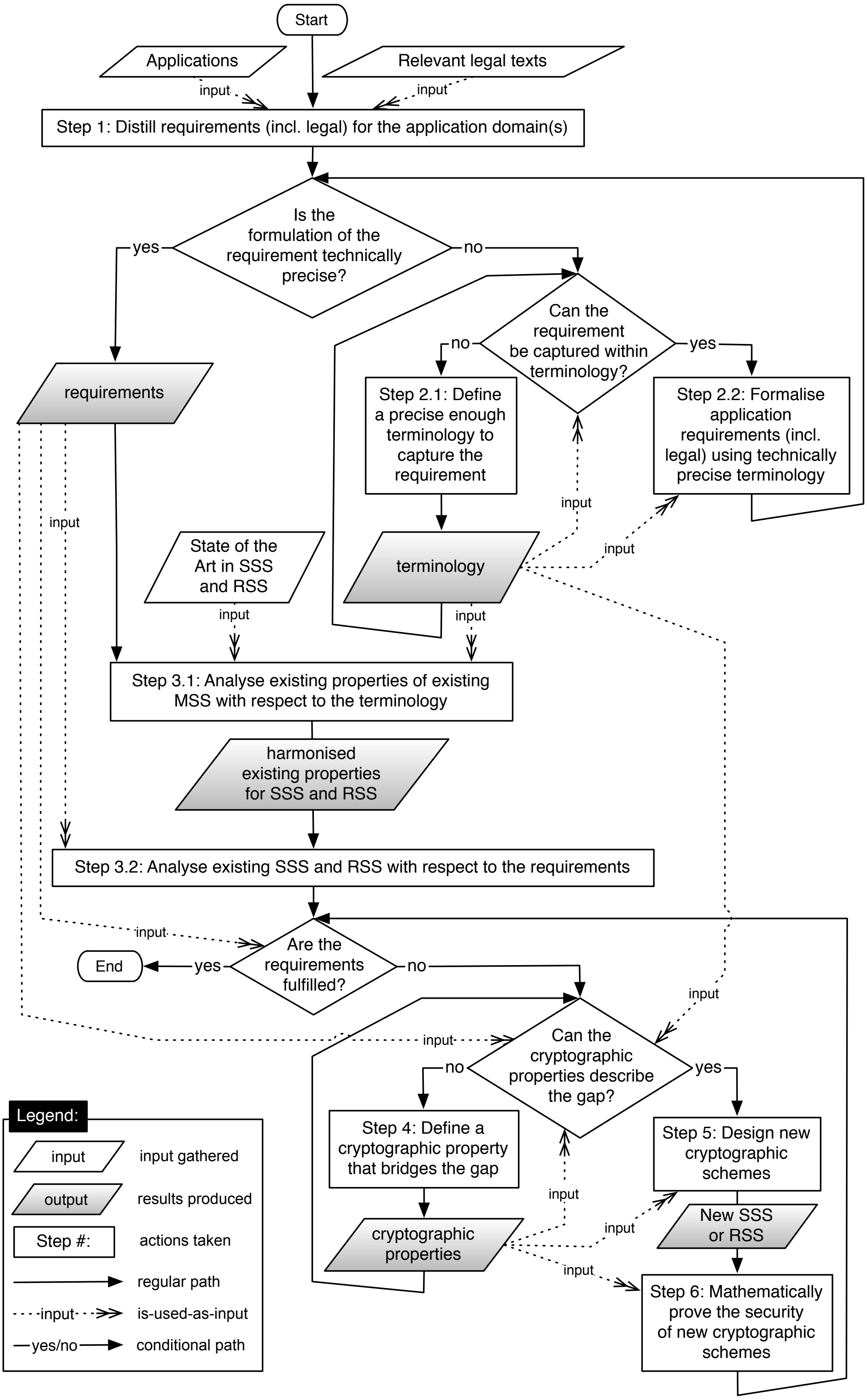

Start
Applications
Relevant legal texts
input
input
Step 1: Distill requirements (incl. legal) for the application domain(s)
Is the formulation of the requirement technically precise?
yes
no
Can the requirement be captured within terminology?
no
yes
requirements
Step 2.1: Define a precise enough terminology to capture the requirement
Step 2.2: Formalise application requirements (incl. legal) using technically precise terminology
input
input
input
State of the Art in SSS and RSS
terminology
input
input
Step 3.1: Analyse existing properties of existing MSS with respect to the terminology
harmonised existing properties for SSS and RSS
Step 3.2: Analyse existing SSS and RSS with respect to the requirements
input
Are the requirements fulfilled?
End
yes
no
input
input
Can the cryptographic properties describe the gap?
no
yes
Step 4: Define a cryptographic property that bridges the gap
Step 5: Design new cryptographic schemes
input
input
cryptographic properties
New SSS or RSS
input
Step 6: Mathematically prove the security of new cryptographic schemes
Legend:
input
input gathered
output
results produced
Step #:
actions taken
regular path
input
is-used-as-input
yes/no
conditional path

Figure 11. The six steps of the methodology and the output of this thesis

While this might confuse or bore readers with a legal background, the thesis hopes to provide a better legal understanding also for computer scientists. Vice versa, technical and cryptographic definitions are likewise restated in plain language, allowing readers with only little cryptographic or information security background to gain an insight. If closely following the literal meaning of the legal texts failed to provide a precise enough definition or if the grammatical interpretation could still be technically misleading, this thesis will adequately use other means of interpretation according to the methods used in studies of law.[68] Helpful in this respect is the *teleological interpretation*: Based on the purpose and goals of the law one extracts the meaning of the legal text[69]. Finally, the analysis will look at other interpretations, e.g., the prevailing opinion stated in literature and, if possible, the prevailing opinion stated by court decisions.

1.6. Organisation of this thesis

The remainder of this thesis is organised as follows into four parts. The first two parts set forth the goals; solutions are in the third part and the evaluation and applications are presented in the final part.

1st Goal: **The cryptographic mechanism for malleable signatures shall award the electronically signed document with a legally high probative value.** This requires to understand and define the legally required technical properties. For the in-depth comparison of existing standard and malleable signature mechanisms, a unifying integrity definition will be formulated. The proposed integrity notion must capture properties relevant for a high probative value and it must capture properties that emanate from explicitly considering the authorization of subsequent modifications. Part I p. 67–187

2nd Goal: **The cryptographic mechanism shall provide confidentiality for the cloaked originals that is legally compliant with data protection requirements and keeps trade secrets confidential.** This — likewise — requires to understand and capture the relevant legal properties in technical terms. It will be checked if the cryptographic notion of privacy formally introduced by *Brzuska et al.* [64] is a suitable requirement for RSS and SSS. Part II p. 189–244

New cryptographic schemes: **The newly designed cryptographic mechanism must meet both goals simultaneously and be practically applicable and cryptographically secure.** The cryptographic properties of existing schemes will be harmonised and then shortcomings will be analysed. New properties for RSS and SSS get introduced to mitigate the identified shortcomings and fulfil the requirements. This thesis presents eight newly devised and provably secure — as defined in Sec. 2.3.1 — malleable signature schemes. Additionally, each scheme is checked for its practical applicability. Practicality is judged by its versatility and flexibility of the policy and if the scheme shows a practically usable execution time on standard hardware, e.g., desktop PC or smart card. Part III p. 245–430

Evaluation: **The increase of the legal probative value for a document signed with a malleable signature scheme that sufficiently protects the integrity and the confidentiality is evaluated.** This part presents the legal argumentation why MSS with the properties defined in this thesis is classified as a qualified electronic signatures under Regulation 910/2014. The newly designed schemes will be evaluated regarding their legal probative value and their compliance as a privacy-enhancing technology (PET) which helps to meet data protection and privacy-by-design requirements. Additionally, applications made possible by private MSS with a high probative value will be briefly presented. Part IV p. 443–540

Before starting with Part I, Chapter 2 and Chapter 3 introduce terms an definitions. Finally, Chapter 19 concludes and presents the contributions and directions for future work.

[68] Techniques, in no particular order, found there are: grammatical interpretation, historical interpretation, systematic interpretation, and teleological interpretation.

[69] For example the 1997 German Digital Signature Act (SigG) has an introduction stating the legislation's purpose in § 1 SigG which has the heading "Zweck und Anwendungsbereich" which could be translated to 'purpose and scope'.

2 —— Existing terms and definitions

Overview of Chapter 2

Chapter 2 states the necessary *existing* terms and definitions that are relevant for this thesis. New terms and definitions, which are the outcome of this thesis, are given in Chapter 3. Alongside, this chapter introduces the notation used in this thesis to describe them.

2.1. Definition of document / message

The thesis sees a message as "recorded information which can be treated as a unit" [256], but it can be further divided and "can contain text, pictures, video and audio content, metadata and other associated content" [256][70].

Definition 1 : Message

This thesis denotes the message by m, where, unless explicitly otherwise noted, $m \in \{0,1\}^*$.

Note 1 The length of the message is finite.

Note 2 For the scope of this thesis the term 'document' can be used interchangeably with 'data' or 'message'.

This thesis will use special symbols to indicate certain conditions as follows:

Definition 2 : Special Symbols

For certain cases this thesis requires special symbols that are not part of the message space $\{0,1\}^*$. $*$, $\blacksquare$ and $\perp$ are examples for special symbols.

The symbol $\perp \notin \{0,1\}^*$ denotes an error or an exception.

The symbol $\blacksquare \notin \{0,1\}^*$ denotes a redaction, visible to Verifiers.

The symbol $* \notin \{0,1\}^*$ denotes a placeholder, i.e., matches all $\{0,1\}^*$ in comparisons.

All special symbols are not in the message space and are different from each other: $* \neq \perp \neq \blacksquare$.

To combine several values, this thesis uses a uniquely reversible concatenation, e.g. a message $m = (m[1], \dots, m[\ell])$. As such, the concatenated m can be treated as one string, e.g. when used as input to an algorithm, however each $m[i]$ for $1 \leq i \leq \ell$ must still remain identifiable and must allow to be recovered individually.

Definition 3 : Uniquely Reversible Concatenation

A value c is constructed by a uniquely reversible concatenation, if concatenating j values $c[1] \dots c[j]$ into c allows at any time to retrieve each individual value $c[i]$ for $1 \leq i \leq j$ from c.

A uniquely reversible concatenation is denoted as "," and "$||$", giving

$$c = (c[1], \dots, c[j]) \qquad or \qquad c = (c[1]|| \dots ||c[j])\,.$$

[70] For further discussion see also this thesis' analysis of ISO-27038 [256] in Sec. 9.3.5.

The length or the number of characters in a string is denoted as follows:

Definition 4 : Length of a parameter

> *The length of parameter x is denoted as $|x|$.*

2.2. Definition of an electronic document

Following the legal texts of Regulation 910/2014, Directive 1999/93/EC, and SigG, this thesis has adopted the legal term electronic signature as follows:

Definition 64 : Electronic Document

> *The electronic document refers to the message, as defined in Definition 2, that carries digitised information when it is referred to in a legal context.*
>
> *Note 1 This is a legal term.*
>
> *Note 2 The information contained in an electronic document is digitised and can be texts, pictures, sounds as well as videos following [46, p. 1016] and [219, p. 10].*

See Sec. 4.2.1 for the background analysis that led to the above definition.

2.3. Definition of cryptographic primitive, cryptographic scheme and cryptographic provable security

This thesis, following cryptographic literature, "divides cryptographic mechanisms into primitives (such as [...] public key primitive and hash functions) and schemes (such as [...] signature schemes [...])" [188].

Definition 5 : Cryptographic Primitive from ENISA

> *"Cryptographic primitives are considered the basic building blocks upon which one needs to make some assumption. This assumption is the level of difficulty of breaking this precise building block; [...]" [188]*

2.3.1. Definition provable security

This thesis is concerned with cryptographic algorithms to provide a level of security that is sufficient to fulfil legal requirements. This thesis follows the work of [38, 154, 160, 210, 211] and other works in the field of MSS that adopted the ideas of provable security. In this thesis the term 'provably secure' is used to indicate that the security is established following the "[...] rigorous analysis methodology of cryptography (which evolves around clear definitions and rigorous inference rules) [...]" [207]. It shall further highlight that a scheme's security is only established against that clearly defined goal and only proven on the cryptographic level (see the discussion on the limits in Sec. 2.3.3). *Goldreich* criticised in that same essay [207], from which the previous quote on rigorous analysis is taken, that "[...] within the domain of a rigorous analysis of cryptography, [...] the adjective "provable" adds nothing to the claim of security [...]"[71] [207]. This thesis still uses the notion for two reasons: First, it was found in works in the area of MSS. Second, it is used in order to make a clear statement that such a cryptographically rigorous proof is offered. The author of this thesis hopes that this will not confuse the readers[72].

This thesis understands and defines provable security as follows:

[71] The full quote from *Goldreich* reads: "Thus within the domain of a rigorous analysis of cryptography, the term "provable security" (and in general "provable property") makes no sense; that is, the adjective "provable" adds nothing to the claim of security (assuming that the claim is valid, and it cannot be applied if the claim is invalid or unknown to be valid)." [207].

[72] *Goldreich* contrary noted that "Indeed, qualifying a noun by an adjective that adds nothing to it is peculiar, but more importantly it may only cause confusion." [207].

Definition 6 : Provably secure

> *An algorithm is provably secure if an attack against the algorithm (denoted as $\mathcal{A}$) can be* ***tightly*** *and* ***efficiently reduced*** *into another algorithm ($\mathcal{B}$) that finds a solution to a problem ($\mathcal{P}$) known to be* ***hard****.*
>
> *'Known to be hard' here means that a solution to the problem is at least "[...] believed to be hard to find [...]" [287]. A problem is hard if finding a solution to the problem is computationally intense and therefore time consuming. The compute power and time needed is related to a security parameter.*
>
> *Increasing the security parameter allows to increase the complexity of the attack. The security parameter, denoted as λ, can comprise several parameters, including what is commonly referred to as key-length. The adversary's probability to successfully mount an attack should be negligible as a function of λ (see Sec. 2.6 for definition of negligible success). Secure choices for those parameters and for hard problems are offered by several guidelines, e.g., [84, 188].*
>
> *A reduction is tight if the runtime for the attack against the known security problem, i.e., $\mathcal{B}$, is only negligibly longer than that of the attack against the algorithm to be proven, i.e., $\mathcal{A}$; and the success probability of $\mathcal{B}$ is also only negligibly lower than that of $\mathcal{A}$.*
>
> *A reduction is efficient when it runs in probabilistic polynomial time (PPT).*

This method is also known under the term "reductionist security", e.g. [402]. It requires as a starting point a formal description of the security property, the new algorithm and a formal description of adversary's capabilities. The latter includes defining what problems are hard to achieve for the usually bounded adversary. This is based on commonly agreed complexity theory following the ideas of *Diffie and Hellman* [154]. [73]

The formal description of the security property is done in this thesis using games, sometimes also called experiments, as defined and described in Sec. 3.8.

EXAMPLE

As an example see the goal to be secure against existential unforgeability under chosen message attacks (EUF-CMA) as defined in Definition 95 taken from *Goldwasser et al.* [211]. The so-called RSA-assumption introduced by *Rivest, Shamir, and Adleman* states that an adversary which is given n where $n = p \cdot q$ with p, q being "very large, "random" primes" [408] will not be able to obtain p and q. The underlying problem is extracting the eth roots modulo a composite integer. "[...] [D]ue to the enormous difficulty in factoring n [...]" [408], which is one obvious solution to this problem and the best currently known, this is considered hard. Under these assumptions it was proven that RSASSA-PSS from PKCS-v2.2 [422] is provably secure with respect to the EUF-CMA property for a sufficiently large security parameter. Currently a size of 2048 Bits for the size of the modulus n is considered hard [84, 188]. This means that the algorithm provably offers security against existential unforgeability under chosen message attacks based on the security of the assumptions stated.

2.3.2. Definition of a provable secure scheme

Schemes are themselves built out of primitives [188]. A scheme's security properties should be formally defined and the scheme should be provably secure with respect to the properties.

Definition 7 : Cryptographic Scheme from ENISA

> *"By a [cryptographic] scheme we mean some method for taking a primitive, or set of primitives, and constructing a cryptographic service out of the primitive." [188]*

Even when built upon provably secure primitives a scheme's security needs to be defined and proven "[...] because a scheme [that] makes use of a secure primitive does not imply the scheme is secure" [188]. The proof is in the form of "[...] an algorithm which takes an adversary against the scheme in some well defined model, and turns the adversary into one which breaks some property of the underlying primitive (or primitives) out of which the scheme is constructed." [188]

[73] For example, a polynomially time bounded adversary can not search for a factorisation of a large numbers with only two prime factors, because the current solutions require more than polynomial time in relation to the size of the number and are thus defined as too time consuming for this kind of adversary.

To simplify proofs, a new cryptographic scheme solely constructed from existing primitive(s) with known security properties can be proven by a reduction to the underlying primitive(s). The new scheme is then shown to be as strong as the underlying primitive(s). In order to achieve this, the proof shows that an adversary that can break a security property of the new scheme would only succeed if the adversary can also gain an advantage in breaking a security property of the said primitive(s).

EXAMPLE

As an example many schemes in this thesis are constructed based on unforgeable signature schemes, for which constructions with security proofs for several hardness assumptions exists, e.g., RSA-assumption [408] or the discrete logarithm problem (DLP) over carefully chosen groups [154, 160]. Then the formal security proof shows that an attack against the unforgeability of a secure MSS can be reduced/transformed into an adversary that breaks the unforgeability of the unforgeable signature scheme used as primitive. Thus, from the use of a provably secure unforgeable signature scheme the security of the MSS is shown to follow.

As above, a primitive's cryptographic security itself is again given by a, at best tight reduction, to a problem which has currently a known computational complexity which is considered to be a "hard problem" [188].

EXAMPLE

As an example for the computational complexity take again the security of the public key primitive based on the RSA-assumption as introduced by *Rivest, Shamir, and Adleman*. In a nutshell this primitive bases its security on the observation that the best solution to finding the *e*th roots modular a composite integer is factorisation that can only be done at an "enormous difficulty" [408] and is currently considered a "hard problem" [188]. Hence, the primitive is *currently* considered secure if the factors are big enough, as solving the factorisation problem takes longer if the size of the factors is bigger. But if quantum computing becomes technically feasible at a certain scale, then factorisation will no longer be a hard problem. [188, 439].[74] Hence, those primitives are not considered post-quantum safe [108]. For other primitives, other mathematically hard problems are the basis for their security.

2.3.3. Provable security proves security on the cryptographic level

This thesis seeks to increase the practicality of the MSS for their use in real life environments, hence it wants their cryptographic strength high enough to use the scheme to uphold a claim in a legal dispute. As such the thesis also considers it "[...] good cryptographic practice to only use schemes for which there is a well defined security proof which reduces the security of the scheme to that of the primitive." [188]. Hence, provable security is shown for all newly devised schemes in this thesis. "If a protocol is proven secure in this way then, as long as the "hard" mathematical problems remain sufficiently hard, the protocol is unbreakable within the defined security model." [402].

Note, the statement of security made by provable security is made against well defined cryptographic properties. It is in particular important to challenge the definitions in order to understand if the defined cryptographic property is indeed matching the real-life application scenario or if it is an "inadequate" or "unsatisfactory definition"[75] [207].

Note further, the statement made by provable security covers only the scope of cryptographic algorithms[76]. Especially it is not making any statements about the security of the implementation or the deployment and use of that algorithm. These additional sources of insecurity are also out of scope of the thesis (see Sec. 1.4.3). For a discussion of limitations and problems see for example *Radke, Boyd, Gonzalez Nieto, and Brereton* [402] who observe that:

[74] The ENISA report correctly notes that "[i]f the development of quantum computers became imminent, then all this documents guidelines would need to be seriously reassessed. In particular all of the public key based primitives in this document should be considered to be insecure" [188]. Increasing the key length of attacked algorithms is not a feasible solution as a much greater overhead is imposed on regular computers then the security gained against the quantum-capable attacker.

[75] *Goldreich* called them as such, stating that "[...] also in these cases, schemes were constructed and shown to satisfy the corresponding [unsatisfactory] definitions. Thus, the point is not proofs (of "security") but rather what is being proved." [207].

[76] See Fig. 12 on page 30 for a pictographic explanation of bridging terminology out of the cryptographic realm.

> "Unfortunately, many protocols so proven to be secure in theory, have been found to be insecure in practice, when deployed in the real world. This inequality between the theoretical security and the actual security can be traced back to a deficiency in the security proof model. The mathematical security models while useful, especially for examining the security of a protocol in isolation, do not take into account the wide range of side channel attacks, social engineering, and interfaces to other protocols and the environment, which occur in the real world." [402]

This thesis provides cryptographic schemes with provable security and properties that are aligned with the legal requirements. This thesis thus assumes that the implementation, deployment and use of the provably secure malleable signature schemes is provided and ensured by other means as necessary. This thesis provide informal descriptions of all security properties and explain the formal game-based property definitions in detail. This allows to follow the thesis' final reasoning that those definitions capture the legal requirements. Hopefully, it also allows to challenge and check their suitability for a new application and for flaws[a].

[a] To one more time use a quote by *Goldreich*: "The march of science: freedom, mistakes, and revisits" [207].

2.4. Notations for algorithmic descriptions

Definition 8 : Value Assignments

> *The input of an algorithm has the values* $(input_1, \dots, input_n)$*. The output is denoted as* $(output_1, \dots, output_n)$*. The assignment from input to output value is expressed by* $\leftarrow$*. The execution of the* Algo *algorithm is denoted as:*
>
> $$(output_1, \dots, output_m) \leftarrow \mathsf{Algo}(input_1, \dots, input_n)\,.$$

2.5. Notations for a random bit

Definition 9 : Uniformly Random Bit

> *To denote a single bit, i.e.,* $b \in \{0,1\}$*, randomly drawn from a uniform distribution over* $\{0,1\}$ *this thesis writes:*
>
> $$b \xleftarrow{\$} \{0,1\}\,.$$

2.6. Definition of an adversary's negligible success and efficient algorithm

Definition 10 : Negligible from *Katz and Lindell*

> *"A function f is negligible in cryptography if for every polynomial $p(.)$, there exists an N such that for all integers $n > N$ it holds that $f(n) < \frac{1}{p(n)}$." [278, Def. 3.4, p. 56]*

This thesis considers algorithms efficient if they can be run in probabilistic polynomial time (abbreviated as PPT):

Definition 11 : Probabilistic Polynomial Time (PPT) from *Katz and Lindell*

> *"An algorithm $\mathcal{A}$ runs in polynomial time if there exists a polynomial p such that, for every input $x \in \{0,1\}^*$, the computation of $\mathcal{A}(x)$ terminates within at most $p([|x|])$ steps. [...] By default, we allow all algorithms to be probabilistic (or randomized) [...] in addition to its input, [it] is given a uniformly distributed random tape of sufficient length whose bits it can use, as needed, throughout its execution."[77] [278, p. 54]*

As *Katz and Lindell* explain, the combination of both of the above terms is suited to define asymptotic security and computational security [278].

[77] The American English spelling from original work is retained for terms and quotes.

2.7. Background on the Random Oracle (RO) and limitations

Rather than a definition this section provides a short high level background into the so called random oracle (RO). The origin is probably the work by *Fiat and Shamir* [192] and a refinement by *Bellare and Rogaway* [37]. In a nutshell, the random oracle model (ROM) assumes that all participating parties have access to an idealised but truly random function. This function is called the random oracle, hence the name. "[...] the idealized random function is instantiated through a "good" cryptographic hash function, like SHA-1 or a variation thereof."[78] [56]. With that the random oracle often makes it "practical" [37] to find an efficient solution and to prove its security. Of course there is a difference between the security achieved by an algorithm provably secure in the standard model and one that is only provably secure in the random oracle model. Apart from the problem to get a truly random number generator in practical environments, schemes secure in the ROM are — just — "[...] usually believed to remain secure in the standard model, proofs in the RO model do not technically guarantee this, but merely provide some evidence of security." [56]. The interested reader is directed to the original works and recent works, like *Boldyreva and Fischlin*'s analysis of the ROM and the OAEP encryption [56].

The contrary is the standard model. In a nutshell, a proof in the standard model only assumes that standard cryptographic hardness assumptions hold. Those assumptions assume that certain problems are hard as discussed in Sec. 2.3.2, e.g. the assumption that factoring is hard.

2.8. Definition of a cryptographic hash function primitive

ISO 10118-1 describes a hash-function as a "[..] function which maps strings of bits to fixed-length strings of bits, satisfying the following two properties: it is computationally infeasible to find for a given output an input which maps to this output; it is computationally infeasible to find for a given input a second input which maps to the same output" [240]. The output is denoted as "hash-code" [240]; ISO 10118-1 explicitly acknowledges other synonymous terms, such as the term 'hash-value' used in this thesis. Additionally, the cryptographic hash functions used in many classic digital signature schemes have been designed with "a well-known heuristic in cryptographic hash function design" [217] called avalanche effect [217, 453], meaning that (a) the hash-value depends on every input bit and (b) that a change of one input bit results in approximately 50% changed bits in the hash-value.

Definition 12 : Cryptographic hash-function

A mathematical function, denoted as $\mathcal{H}$, that maps its parameter from the domain $\mathcal{S}$ into the hash domain $\mathrm{dom}(\mathcal{H})$.

The output is referred to as hash-value[79] *in this thesis. The function is compressing, i.e., the hash-value's domain's size must be smaller or equal to that of the input domain: $|\mathcal{S}| \geq |\mathrm{dom}(\mathcal{H})|$. Cryptographic hash-functions are a subset of all hash-functions. A cryptographic hash-function must further be non-keyed, collision-resistant, and one-way.*

A cryptographic hash-function is a compressing, non-keyed, collision-resistant, and one-way function $\mathcal{H}$ which maps any arbitrary string of bits $m \in \{0,1\}^*$ onto a string of bits (the hash-value) of a fixed width τ:

$$\{0,1\}^\tau \leftarrow \mathcal{H}(m)\,.$$

The size of the output (hash-value) is the security parameter denoted as τ.

[78] The American English spelling from original work is retained for terms and quotes.

[79] ISO 10118-1 defines the term 'hash-code', which it considers synonymous to 'hash-value', as the "string of bits that is the output of a hash function" [240].

2.9. Definition of a *classic* digital signature scheme based on asymmetric cryptography (CDSS)

This thesis will use the term "electronic signature" to refer to the legal notion and the term *digital signature scheme* (DSS) to refer to an cryptographic algorithm that creates a digital signature, which is denoted cryptographically by σ. The signature schemes in this thesis are constructed on asymmetric cryptographic techniques, generally termed *asymmetric signature systems* following ISO 9798 [242]. Further, this thesis will call a DSS without malleability a *classic digital signature scheme* (CDSS) to differentiate them from malleable signature schemes (MSS).

2.9.1. Definition of an asymmetric signature system following ISO 9798

The following definition is taken from ISO 9798 [242], which sets it into the context of a generally defined *asymmetric cryptographic technique*. The latter is a "cryptographic technique that uses two related transformations, a public transformation (defined by the public key) and a private transformation (defined by the private key). The two transformations have the property that, given the public transformation, it is computationally infeasible to derive the private transformation." [242].

Definition 13 : Asymmetric Signature System from ISO 9798 [242]

> *"**asymmetric signature system:** a system based on asymmetric cryptographic techniques whose private transformation is used for signing and whose public transformation is used for verification." [242]*

2.9.2. Algorithmic description of a CDSS

The following definition of a CDSS is based partly on the ITU X.800's definition of a *digital signature mechanism* [263].

Definition 14 : Classic Digital Signature Scheme (S) and Signature (σ)

A classic digital signature scheme S consists of three algorithms with polynomial runtime (PPT):

$$S = (KGen_{sig}, Sign, Verify)\,.$$

All three algorithms get the security parameter λ as an additional input.

Note, all algorithms may output $\bot$ in case of an error.

Key Generation: *Digital signature schemes are based on an asymmetric key pair, denoted as (sk_{sig}, pk_{sig}), which is generated by the algorithm $KGen_{sig}$.*

The key generation algorithm $KGen_{sig}$ creates one pair of keys for the Signer, which consists of a private signature generation key pk_{sig} and the corresponding public verification key pk_{sig}, where λ denotes the security parameter:

$$(pk_{sig}, sk_{sig}) \leftarrow KGen_{sig}(1^{\lambda})\,.$$

The private key sk_{sig} shall be private, i.e. "can only be used by that entity"[247]. Hence, pk_{sig} is

(a) kept confidential,

(b) kept under the Signer's sole control (i.e., can only be used by Signer), and

(c) unique to the Signer (i.e., together with a chain of public key certificates).

The verification key or public key pk_{sig} "can be made public"[247], e.g. as part of a public key certificate (see Section 2.11).

Signing: *The entity called Signer uses the algorithm Sign to generate the signature σ over a message m. Sign uses the private signature generation key sk_{sig}.*

The Sign algorithm takes as input the security parameter λ, a message m with $m \in \{0,1\}^*$, and the signer's secret key sk_{sig}. It outputs a signature σ and the message (or $\perp$ on error):

$$(m, \sigma) \leftarrow \mathsf{Sign}(1^\lambda, m, \mathsf{sk}_{sig})\,.$$

Verification: *A digital signature σ can be verified using the algorithm Verify by any entity called Verifier holding the message-to-be-verified, the signature-to-be-verified and the signer's public verification key pk_{sig}. A positive verification result (`true`) corroborates that the signature σ was produced with the Signer's private signature generation key sk_{sig} corresponding to the Signer's verification key pk_{sig}.*

The Verify algorithm takes as input the security parameter λ, the message m, the signature σ, and the public verification key pk_{sig}. It outputs a decision as a bit $d \in \{\texttt{false}, \texttt{true}\}$, indicating the validity of the signature σ for the message m with respect to the Signer's public key pk_{sig}.

$$d \leftarrow \mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig})\,.$$

Transport of keys and trust in keys is out-of-scope of the algorithms. The transport of the public verification key $\mathsf{pk}_{\mathrm{sig}}$, or the signature σ and message m, from the Signer to the Verifier is out of scope of this thesis. Hence, it is also not within the scope of the definition of a Digital Signature Scheme (DSS). However, the Verifier **is required to have knowledge of** the public verification key $\mathsf{pk}_{\mathrm{sig}}$ to verify the digital signature. Note further that the Verifier **is required to have trust in** the binding of the public verification key $\mathsf{pk}_{\mathrm{sig}}$ to an identifier of the communicating entity in order to obtain origin authentication.

2.9.3. Definition of correctness for DSS

This thesis assumes the three algorithms noted above in Section 2.9.2 to work correctly. Hence, a signature generated using S's signature generation algorithm, which this thesis denotes as S.Sign, shall verify using S's signature verification algorithm S.Verify, if the corresponding keys created by $\mathsf{S.KGen}_{\mathrm{sig}}$ are involved.

Definition 15 : Signing Correctness following *Brzuska et al.*

For any security parameter λ, any key pair $(\mathsf{sk}_{sig}, \mathsf{pk}_{sig}) \leftarrow \mathsf{S.KGen}_{sig}(1^\lambda)$, any message $m \in \{0,1\}^*$, and any $\sigma \leftarrow \mathsf{S.Sign}(1^\lambda, m, \mathsf{sk}_{sig})$ the following holds:

$$\mathsf{S.Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}) = \texttt{true}\,.$$

The above correctness assumption for the standard signature scheme has been adapted from the one given for SSS by *Brzuska et al.* [64] (see Sec. 11.4).

2.9.4. Definition of cryptographic unforgeability for DSS

According to *Goldreich* a digital signature scheme (DSS) shall "satisfy the following:

- each user can efficiently produce his own signature on documents of his choice;
- every user can efficiently verify whether a given string is a signature of another (specific) user on a specific document; but
- it is infeasible to produce signatures of other users to documents they did not sign" [208].

The authors then define unforgeability in [208] as follows:

Definition 16 : Unforgeability from *Goldreich*

> *"A digital signature scheme is [...] unforgeable [...] if every feasible chosen message attack succeeds with at most negligible probability." [208]*

Underlying to *Goldreich*'s definition is that unforgeability shall withstand the attack of generating an existential forgery under an adaptive chosen message attack (EF-CMA) as defined by *Goldwasser, Micali, and Rivest* [211]. This can be described on a high level as follows:

Definition 17 : Existential unforgeability under an adaptive chosen message attack following *Goldwasser et al.*

> *The adversary can use a signing oracle to generate valid signatures on messages of the adversary's choice. To succeed and break unforgeability, the adversary needs to generate a fresh message that verifies under a previously fixed public verification key with non-negligible probability. A message is fresh if the adversary never queried the signing oracle for this message.*
>
> *Note, the forged message does not need to make sense to the application that uses these messages.*

More details on the terms defined by *Goldwasser, Micali, and Rivest* and a formal definition of Definition 17 can be found in Sec. 5.3.4.

2.10. Definition of involved entities to bridge cryptographic and legal terminology: Signatory, Signer, Verifier, Certificate Issuer

Following the cryptographic Definition 14, the generation of the digital signature involves algorithms that generate or require certain keys, i.e., from the key-pair $(\mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{sig}})$ the signature generation algorithm Sign requires the input of $\mathsf{sk}_{\mathrm{sig}}$. Using the access to the secret key material, like $\mathsf{sk}_{\mathrm{sig}}$, as a differentiator, the thesis differentiates four non-exclusive roles that an entity can have:

- Signer and Certificate Issuer all know their own secret keys;
- Verifier does not need to know any secret keys.

An entity does technically not need to know the actual contents of the secrets in order to facilitate them, i.e., the secret $\mathsf{sk}_{\mathrm{sig}}$ could stay confined in an hardware security module (HSM) as defined in Definition 29; then the Signer technically becomes the entity that can and must authorize the use of $\mathsf{sk}_{\mathrm{sig}}$ inside the HSM.

Definition 18 : Signer in DSS

> *The entity denoted as Signer can use the secret signing key sk_{sig}. Thus, the Signer can use sk_{sig} as input to algorithm Sign to generate signatures that verify under pk_{sig}. The Signer does not know and thus cannot use sk_{TTP}.*

In Fig. 12 these three roles are presented as an intermediate layer bridging the cryptographic definition and the legal terms.

The entity in the role denoted Signer can be aligned with the entity described in legal terms as the *holder of the signature key*[80]: In the superseded German SigG the entity has been described as the natural person that possesses[81] the cryptographic signature key[82], which is legally denoted as "electronic signature creation data" [Regulation 910/2014]. In other texts this role is also termed *issuer*[83]; however this term is not used as it is already in use for a different role in Regulation 910/2014[84]. This thesis

[80] In German it was termed "Signaturschlüssel-Inhaber" [§ 371a (1) ZPO a.F.] in an older version before Regulation 910/2014 and is now "verantwortende Person" [§ 371a (1) ZPO n.F.].

[81] In the German original text it is defined as follows: "[...] "Signaturschlüssel-Inhaber" natürliche Person, die Signaturschlüssel besitzen; [...]" [§ 2 (9) SigG]; 'besitzen' has been translated as 'possess' according to Langenscheidt Recht Englisch [70].

[82] In German termed "Signaturschlüssel" [SigG].

[83] Can be translated as 'issuer' from the German "Aussteller" [§ 126a (1) BGB].

[84] "[...] issuer or issuers of electronic identification [...]" [Art. 9 Regulation 910/2014].

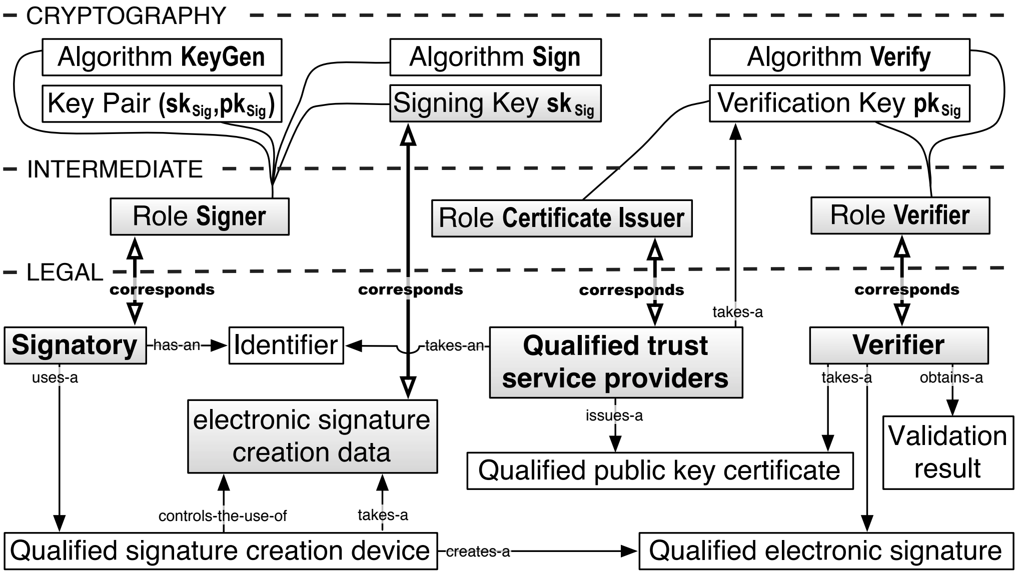


Figure 12. Mapping cryptographic definition of a digital signature to legal terminology from Regulation 910/2014 (eIDAS): Cryptography sees it as three algorithms and a key-pair, legal terms are centred around the signatory; the three roles aid bridging the two 'worlds'; in this figure the emphasise is put on terms connected to the natural person in the role of the Signer.

follows Regulation 910/2014 and uses the term "signatory" [Art. 3 (9) Regulation 910/2014][85]. This thesis makes the following alignments between legal and cryptographic terminology: Signatory corresponds to Signer and $\mathsf{pk}_{\mathrm{sig}}$ corresponds to "signature creation data" [Regulation 910/2014]. This and other correspondences are depicted in Fig. 12. This is captured in the following definition:

Definition 19 : Signatory in DSS following Regulation 910/2014

> *The term " 'signatory' means a natural person who creates an electronic signature;" [Art. 3 (9) Regulation 910/2014]. The legal entity that is in the technical role of the Signer as defined in Definition 18 will be denoted as signatory. The signatory can use[86] the "electronic signature creation data" [Regulation 910/2014], i.e. the secret signing key sk_{sig}. Thus, the signatory is assumed to become linked to the technical signature σ the signatory creates using sk_{sig}, among other, as input to algorithm Sign. The signatory does not know and thus cannot use sk_{TTP} to issue a public key certificate.*

Henceforth, the term signatory is used to denote the natural person (as defined in Sec. 2.12) that is linked to the technical entity acting as Signer. Legally, a natural person, as defined in Sec. 2.12, is carrying out actions in the real world. In this thesis the term signatory denotes the natural person who is acting as the Signer. To align both terms, a linkage between the natural person and the technical identifier, which is the public signature verification key ($\mathsf{pk}_{\mathrm{sig}}$), must be provided. Henceforth, this thesis assumes additional organisational and technical measures that provide linkage between $\mathsf{pk}_{\mathrm{sig}}$, $\mathsf{sk}_{\mathrm{sig}}$ and the legal person.

From a technical perspective, the term Signer describes the same party as the following legal terms:
- "signatory" [Art. 3 (9) Regulation 910/2014][a],
- "Aussteller"[b] [§ 126a (1) BGB], and
- "Signaturschlüssel-Inhaber"[c] [§ 371a (1) ZPO a.F., § 2 (9) SigG], and
- "verantwortende Person" or the "person responsible"[d] [§ 371a (1) ZPO n.F.].

[85] The term 'signatory' is also used in the domain of International Trade Procedures described by the United Nations [504].

[86] Note, the Signer does neither technically nor organisationally need to know the secret, e.g., only authorize the secret key's use inside a hardware security module following the requirements prescribed for a qualified signature-creation device (QSCD) (see Sec. 4.4.7).

Note, the establishment of the link between legal person and cryptographic key, as well as the organisational and technical means to achieve this link, and finally key distribution are **out of scope** of this thesis (see Sec. 1.4.3). Such a link can be established by a Certificate Issuer that issues a public key certificate as defined in Definition 26. The Certificate Issuer needs to be trusted by the Verifier, as the Verifier entrusts the Certificate Issuer to check the link between the natural person's identifier and the $\mathsf{pk}_{\mathrm{sig}}$ and add both into the certificate as defined in Definition 26.

[a] Same term was found in the former Directive 1999/93/EC.

[b] However, this would more directly translate to 'issuer'; the term 'issuer' is already in use for a different role in Regulation 910/2014: "[...] issuer or issuers of electronic identification [...]" [Art. 9 Regulation 910/2014].

[c] This translates to 'holder of the signature key'.

[d] This translation follows the translation of § 130a sentence 2 provided for ZPO a.F..

Other entities, not knowing $\mathsf{sk}_{\mathrm{sig}}$, are considered being able to assume the role of a Verifier if they participate. This thesis sees any entity to have at least one of the following roles when participating in a DSS: Signer, Verifier, or Certificate Issuer. Some roles are categorised by needing knowledge of a certain secret, some by the absence of this need. The adversary is not considered as separate role and is described in Sec. 3.8.

Definition 20 : Overview of Involved Entities in DSS

Signer

- *requires knowledge of the secret signing key* sk_{sig}*;*
- *does not know* sk_{TTP}*;*
- *generates signature* σ *using* sk_{sig}*, among others, as input to algorithm* Sign*.*

Verifier

- *does not know* sk_{sig} *or* sk_{TTP}*;*
- *verifies signature* σ *using only public information, e.g.,* pk_{sig} *and* pk_{TTP}*.*

Certificate Issuer (as a trusted third party)

- *requires knowledge of the secret certificate issuing key* sk_{TTP}*;*
- *does not know* sk_{sig}*;*
- *issues certificates binding an* `id` *string to the public key (e.g., knows* pk_{sig}*);*
- *is trusted by all parties.*

2.10.1. Workflow of a DSS and definition of a trusted third party (TTP)

The workflow for the generation and verification of a signature is depicted in Fig. 15. The figure also includes the trusted third party (TTP), denoted as Certificate Issuer, which is involved in generating the trust in public verification keys, as depicted in Fig. 14. A digital signature allows the Verifier to verify — by the use of asymmetric cryptographic techniques (see Sec. 2.9.1) — that the signing operation involved the signing key that cryptographically corresponds to the public verification key used in the verification algorithm. A correct (see Sec. 2.9.3) and unforgeable (see Sec. 2.9.4) signature scheme enables the Verifier to corroborate that only the entity that holds the signing key could have generated a valid signature. However, signature verification does not allow to do more, e.g., assert the trust level of that entity, without the use of protocols or the help of third parties. The Certificate Issuer in Fig. 14 could be seen as a certification authority (CA) of an existing public key infrastructure (PKI) [162, 220, 236, 282], that issued a public key certificate to assure the bond of the Signer's identifier, i.e., a bitstring, to the Signer's public key.

Following the separation of *Menezes et al.*, this thesis assumes the trusted third party (TTP) to be functionally trusted:

Definition 21 : Functionally trusted TTP by *Menezes et al.*

> *"A trusted third party (TTP) is said to be functionally trusted if the entity is assumed to be honest and fair but it does not have access to the secret or private keys of users." [337]*

This thesis also assumes that the trusted third party is only involved off-line. This means that the Certificate Issuer is not involved in the signature generation or signature verification in real time. The Definition 22 by *Menezes, Oorschot, Vanstone, and Rivest* [337] captures this well; it is depicted in Fig. 13.

Definition 22 : Off-line involved TTP from *Menezes et al.*

> *"off-line: T[TP] is not involved in the protocol in real-time, but prepares information a priori, which is available to A or B or both and used during protocol execution."*[87] *[337]*

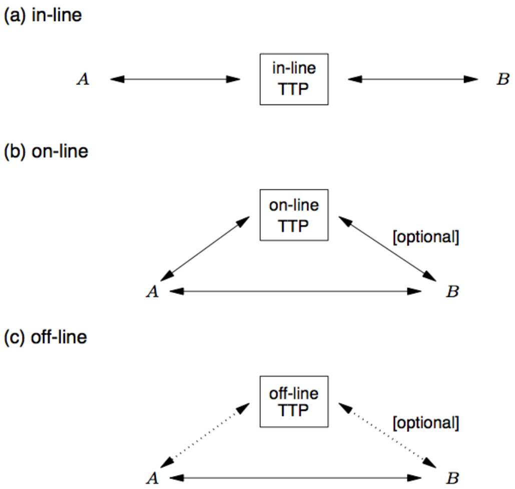


Figure 13. (a) In-line, (b) on-line, and (c) off-line third parties by [337]; dotted lines denote communications carried out prior to protocol run; same information can also be found in ISO 14516 [244, p. 4–5]

Trust establishment for signature verification keys is done through an off-line and functionally trusted third party, but this is out of the DSS's scope

The signature scheme is assuming that the Verifier has the correct signature verification key for the entity that signed the document. "A verifier requires assurance that the public key to be used to verify a signature belongs to the entity that claims to have generated a digital signature (i.e., the claimed signatory). That is, a verifier requires assurance that the signatory is the actual owner of the public/private key pair used to generate and verify a digital signature. A binding of an owner's identity and the owner's public key shall be effected in order to provide this assurance." [355, p. 10] Only with this assurance entity authentication of the Signer is achievable[a].

The additionally required technical and organisational methods are out of the scope of the digital signature scheme. It is assumed to be done through an off-line and functionally trusted third party. Hence, this is also out of scope of all the malleable signature schemes discussed in this thesis.

[a] Namely FIPS states that "[...] the verifier has assurance of the data's integrity, but source authentication is lacking [...]" [355, p. 10]

This thesis assumes that the trust establishment for and distribution of digital signature verification keys is achieved through the help of an off-line TTP. That TTP needs to be functionally trusted for the establishment of a trusted relation between a system entity (e.g., a unique identifier) and its public verification key. This thesis assumes Certificate Issuer being such a functionally trusted, off-line third party and that it generates public key certificates (see Section 2.11) for the involved entities. Fig. 15 shows that the Certificate Issuer is off-line, as it is not involved in the generation and verification of signatures; it additionally shows that the Certificate Issuer is functionally trusted, because it does not need to have access to the signature generation keys, but just to the public signature verification keys.

[87] In Definition 22 the original "T" from *Menezes et al.* was used for the party; in the setting of signatures this thesis termed the entity in that role Certificate Issuer.

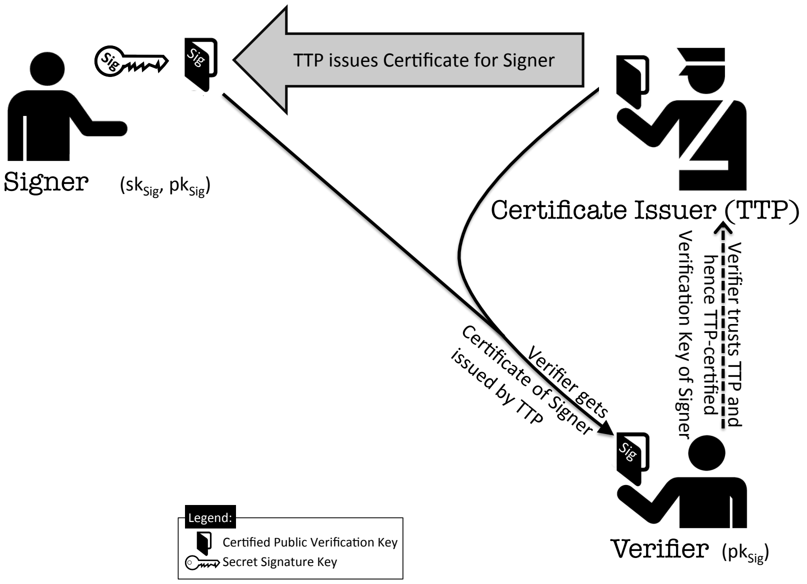

TTP issues Certificate for Signer
Sig
Signer
(sk$_{Sig}$, pk$_{Sig}$)
Certificate Issuer (TTP)
Verifier trusts TTP and hence TTP-certified Verification Key of Signer
Verifier gets Certificate of Signer issued by TTp
Sig
Verifier
(pk$_{Sig}$)
Legend:
Certified Public Verification Key
Secret Signature Key

Figure 14. Trust (dashed line) and verification key distribution in the classic signature process; Initially the signer's verification key is certified once involving the trusted third party

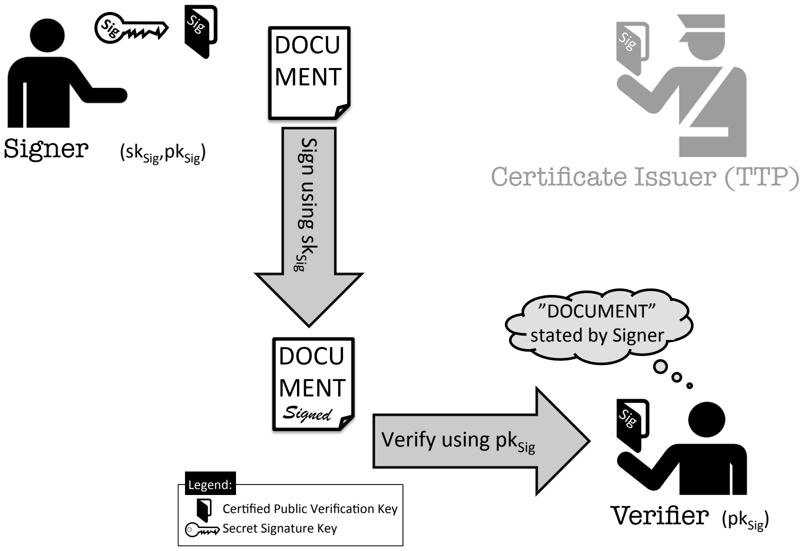


Figure 15. Workflow between the involved parties in the classic signature process; Certificate Issuer as a TTP is off-line and not involved in signature generation and verification

2.10.2. Definition of accountability and Signer-accountability in a DSS

The definition of *Huff* [237] given in 1981 is prominently cited in RFC 4949 [438]. *Huff* restricts accountability to take only "violations or attempted violations of system security" [237] into account[88]. In the case of digital signature schemes that do not allow any subsequent modifications, this notion of accountability still makes sense as all modifications are classified as violations. Hence, all subsequent modifications on a signed document would liberate the Signer from accountability. If in general accountability shall conjointly also authorize subsequent modifications, then accountability must be able to capture those actions regardless of them being no violations. The restriction to violations was also removed from the wording of [237] in RFC 4949 [438]. ISO 13888-1 removed it as well and states

[88] "Accountability. The property that enables violations or attempted violations of system security to be traced to individuals who may then be held responsible." [237]

that accountability is the "property that ensures that the actions of an entity may be traced uniquely to the entity" [243]. However, compared to RFC 4949 that said traced entity is not explicitly said to simultaneously be held responsible. This thesis wants to make explicit that technical accountability aims to enable taking actions against the identified entity. Hence, this thesis will use the definition of accountability from RFC 4949:

Definition 23 : Accountability in General from RFC 4949

> *"The property of a system or system resource that ensures that the actions of a system entity may be traced uniquely to that entity, which can then be held responsible for its actions." [438]*

Henceforth, in the field of digital signatures, accountability means that a signed message, for which integrity is still valid, shall be attributed to the Signer.

2.10.2.1. Non-repudiation of signature generation

To withstand a rebuttal by the accused party this attribution should be backed by technical evidence. "Non-repudiation services establish evidence; evidence establishes accountability regarding a particular event or action." [243, p. V] Following ISO 13888 this can be achieved by means of asymmetric cryptography[89]. The evidence produced by the digital signature scheme needs to be cryptographically strong. This sub-property is called non-repudiation. See Sec. 5.2 for more details on technical non-repudiation definitions.

The achievement of the non-repudiation property by digital signatures is a primary advantage over other integrity protection services. By this property, a Signer is prevented from signing a document and subsequently successfully deny having done so. A non-repudiation service requires specification of precise details including the following:

- an adjudication process,
- an adjudicator (Judge),
- what evidence would be submitted to the adjudicator, and
- what precise process the adjudicator is to follow to render judgement on disputes.

The role of an adjudicator is distinct from that of the Signer, but also from that of a timestamp agent or notary, all of them generate technical evidence that the adjudicator can use.

This is also in line with the general definition that can be extracted from ISO 13888-1 [243] as follows:

Definition 88 : Non-Repudiation Service following ISO 13888-1

> *"[...] [A] non-repudiation service [...] generate[s] [...] and verif[ies] evidence concerning a claimed event or action in order to resolve disputes about the occurrence or non occurrence of the event or action." [243]*

Following the discussion in Sec. 5.2.4 on page 120 the above Definition 88 was selected as a suitable definition to generally describe the aim of the non-repudiation property.

2.10.2.2. 3^{rd}-party verifiability

Apart from the ability to achieve non-repudiation another sub-property of accountability is the distinction whether or not the evidence generated at one time allows not only the Verifier but also third parties to uniquely identify the responsible entity. This thesis calls this 3^{rd}-party verifiability and defines it, taking ideas for wording from [456] as follows:

[89] Part 3 of ISO-13888 is on mechanisms using asymmetric cryptographic techniques (see Sec. 2.9.1 on page 27)

Definition 24 : (Unrestricted) 3^{rd}-party verifiability

> *3^{rd}-party verifiability guarantees that on receipt of a verifying evidence the verifying party can easily and publicly disseminate the evidence and convince an unlimited number of third-party verifiers about the truth of the evidence. The above must work without the verifying party's need to disclose any secrets of the initial verifier or generator. Third-party in the above refers to any party other than the creator of the evidence or the initial verifier of the evidence.*

Note, just to make it explicit, the assumption in this thesis is that the verifiability is not limited, e.g., not restricted in the time, or number of verification. It is out of scope to discuss suitable restrictions and their constructions in this thesis. Henceforth, the thesis will only use the term 3^{rd}-party verifiability while still referring to this unrestricted concept for brevity.

2.10.2.3. Designated verifiability is out of scope

A notion related to 3^{rd}-party verifiability is designated verifiability as offered by designated verifier signature schemes, e.g. [313, 456, 497]. Contrary to a third-party verifiable signature, a designated verifier signature scheme "prevent[s] a designated-verifier from using the DV [designated verifier] signature σ_{dv} on a message m to produce evidence which convinces a third-party that the message m was signed by the signer." [456]. *Steinfeld et al.* further note that chameleon signatures like [292], which are a special way of constructing SSS, "allow designation of signatures to verifiers by the signer" [456, p. 3]. Moreover this property can also be reached for RSS, while preserving RSS functionality, as presented by *Derler et al.* [145]. However, this thesis' focus rests on generating widely usable evidence. Hence, 3^{rd}-party verifiability is of prior interest. Thus, schemes with only designated verifiability are considered out of scope, but the specific legal implications of designated verifier signatures remain of interest and are considered as future work (see Sec. 19.3).

2.10.2.4. Definition of non-repudiable third-party verifiable Signer-accountability in a DSS

Following the above discussion this thesis gives the following notion of Signer-accountability which includes non-repudiation towards an adjudicator:

Definition 25 : Signer-accountability

> ***Signer-accountability*** *allows a predefined set of entities to determine the* Signer *as the origin of a valid signature according to a protocol, i.e., decide that the* Signer *is accountable for the signature of a given valid message-signature pair (m, σ). The valid signature provides electronic evidence that the* Signer *signed the protected message towards the* Verifier *that can later be used by the* Verifier *to convince an adjudicator.*

In the course of this thesis the notion of accountability is further refined, especially as part of the refined integrity notion in Sec. 6.4. Further it is also detailed to work on the scope of individual block of the message in Sec. 13.2.2.

2.11. Definition of a qualified public key certificate

The X.509 standard defines "a framework for obtaining and trusting a public key of an entity in order [...] to verify the digital signature of that entity" [248]. It contains the notion of a *public-key certificate*, which "certifies the binding between the public-key material and the subject of the certificate" [248]. The certificate itself is a data structure that holds, according to Annex I of the previously valid Directive 1999/93/EC, at least the following information:

> "Qualified certificates must contain:
> (a) an indication that the certificate is issued as a qualified certificate;
> (b) the identification of the certification-service-provider and the State in which it is established;
> (c) the name of the signatory or a pseudonym, which shall be identified as such;
> [...]

> (e) signature-verification data which correspond to signature-creation data under the control of the signatory;
> (f) an indication of the beginning and end of the period of validity of the certificate;
> (g) the identity code of the certificate;
> (h) the advanced electronic signature of the certification-service-provider issuing it;
> [...] " [Annex I of Directive 1999/93/EC]

Note, Regulation 910/2014 — superseding Directive 1999/93/EC — amended and rearranged the previous Directive 1999/93/EC Annex I, which served for many years as the EU reference text for electronic signatures, with some more details:

> "Qualified certificates for electronic signatures shall contain:
> (a) an indication, at least in a form suitable for automated processing, that the certificate has been issued as a qualified certificate for electronic signature;
> (b) a set of data unambiguously representing the qualified trust service provider issuing the qualified certificates including at least, the Member State in which that provider is established and:
>
> - for a legal person: the name and, where applicable, registration number as stated in the official records,
> - for a natural person: the person's name;
>
> (c) at least the name of the signatory, or a pseudonym; if a pseudonym is used, it shall be clearly indicated;
> (d) electronic signature validation data that corresponds to the electronic signature creation data;
> (e) details of the beginning and end of the certificate's period of validity;
> (f) the certificate identity code, which must be unique for the qualified trust service provider;
> (h) the advanced electronic signature or advanced electronic seal of the issuing qualified trust service provider;
> [...] " [Annex I of Regulation 910/2014]

The impact of the changes introduced in Regulation 910/2014 on the probative value are accessed in Sec. 7.4 of this thesis. While these changes over the previous Directive 1999/93/EC are affecting the creation, issuing and handling of certificates, the following remains: A public key certificate must exist to fulfil the legal requirement of binding the electronic signature to a natural person which is done by means of the asymmetric key pair[90]. Thus, the main information contained in the public key certificate is the public key of – and an identifier for the natural person. To certify the binding of those two information and induce trust, the certificate is itself signed by a trusted source. "Such a source, called a Certification Authority (CA), certifies a public key by issuing a public-key certificate which binds the public-key to the entity which holds the corresponding private-key." [248]. The Certification Authority (CA), termed *qualified trust service provider* (TSP) in Regulation 910/2014, issues such a public key certificate. This thesis assumes that it operates in a legally qualified manner, which means that it is at least in accordance with Annex II Regulation 910/2014. It reads as follows:

> "Certification-service-providers must:
> (a) demonstrate the reliability necessary for providing certification services;
> (b) ensure the operation of a prompt and secure directory and a secure and immediate revocation service;
> (c) ensure that the date and time when a certificate is issued or revoked can be determined precisely;
> (d) verify, by appropriate means in accordance with national law, the identity and, if applicable, any specific attributes of the person to which a qualified certificate is issued;
>
> [...]
> (g) take measures against forgery of certificates, and, in cases where the certification-service-provider generates signature-creation data, guarantee confidentiality during the process of generating such data;

[90] See Sec. 2.9.1 for a definition of asymmetric cryptographic techniques.

> [...]" [Annex II of Directive 1999/93/EC]

According to the previous European signature legislation (EU Directive 1999/93/EC [175]) and its successor and current legislation (EU Regulation 910/2014 [182]), and the technically most used standard for digital certificates (ITU X.509 [248]) this thesis understands the term qualified certificate or qualified public key certificate as follows:

Definition 26 : Qualified (Public Key) Certificate following Regulation 910/2014 (and Directive 1999/93/EC respectively)

> *A qualified certificate is a digitally signed agreed data structure, such as ITU X.509 [248], that provides the means of an electronic attestation which links signature-verification data (i.e., an entity's public key) to an identifier for a legal entity (i.e., certificate identity code and person's name). The data structure contains at least the data laid down in Annex I Regulation 910/2014 (and Directive 1999/93/EC respectively). For a qualified certificate the issuer is a qualified trust service provider following Regulation 910/2014 who fulfils the requirements laid down in Article 24 Regulation 910/2014 (and a certification-service-provider following Annex II Directive 1999/93/EC respectively).*

2.12. Definition of a natural person and a legal person

The following descriptive definitions of the legal concepts of a natural and a legal person thesis are taken from a guidance document by the *United Kingdom Food Standard Agency (FSA)* [477].

Definition 27 : natural person (legal)

> *"A natural person is a human being, (as opposed to an artificial, legal or juristic person, i.e., an organization that the law treats for some purposes as if it were a person distinct from its members or owner)." [477]*

Definition 28 : legal person (legal)

> *"A legal person has a legal name and has rights, protections, privileges, responsibilities, and liabilities under law, just as natural persons (humans) do. Legal personality allows one or more natural persons to act as a single entity (a composite person - considered under law separately from its individual members or shareholders) for legal purposes." [477]*

2.13. Definition of hardware security module (HSM)

Definition 29 : Hardware Security Module (HSM)

> *In this thesis, the term hardware security module (HSM) is used to refer to a module that can securely keep the secret keys inside the tamper-resistance module during the computation of the required cryptographic algorithms, i.e., the cryptographic functions that need access to the secrets need to run inside/on the HSM. Taken from [17], the term is defined as follows:*
>
> *"[...] an HSM (or Hardware Security Module) is [...] a piece of hardware and associated software/firmware that usually attaches to the inside of a PC or server and provides at least the minimum of cryptographic functions. These functions include (but are not limited to) encryption, decryption, key generation, and hashing. The physical device offers some level of physical tamper-resistance [...]*
>
> *To be optimally secure, the HSM should store all of the keys on the physical device itself. [...]" [17]*

In the domain of digital signatures, these were legally known as secure signature-creation device (SSCD) and following the new Regulation 910/2014 they are termed qualified signature-creation device (QSCD) (see Sec. 4.4.5 and Sec. 4.4.7 for a legal introduction); and the modules distributed to the user as a natural person are often secure smart cards [403].

2.14. Definition of (qualified) electronic signature

The current legal texts on electronic signature in the EU (Directive 1999/93/EC) as well as the future Regulation on electronic signature in the EU (Regulation 910/2014) further define terms that relate to the terms signatory, as defined in accordance with Regulation 910/2014 in Definition 19, and qualified electronic signature. This thesis will use them according to their definitions in Article 3 Regulation 910/2014. Namely, the term qualified electronic signature is built upon two other terms as follows:

Definition 30 : Electronic signature from Regulation 910/2014

> *" 'electronic signature' means data in electronic form which is attached to or logically associated with other data in electronic form and which is used by the signatory to sign;" [Art. 3 (10) Regulation 910/2014]*

Definition 31 : Advanced electronic signature from Regulation 910/2014

> *" 'advanced electronic signature' means an electronic signature which meets the requirements set out in Article 26;" [Art. 3 (11) Regulation 910/2014]*

The details of Article 26 Regulation 910/2014 are not restated here, they get stated in Sec. 4.4.6 on page 91.

Definition 32 : Qualified electronic signature from Regulation 910/2014

> *" 'qualified electronic signature' means an advanced electronic signature that is created by a qualified electronic signature creation device, and which is based on a qualified certificate for electronic signatures;" [Art. 3 (12) of Regulation 910/2014]*

2.15. Definition of personally identifying information (PII), personal data and data subject

More information on the background is given in Sec. 8.2.3 and the following definition is re-stated from Sec. 8.4.1. It is taken from the Regulation (EU) 2016/679 which is known as the EU's General Data Protection Regulation, or GDPR in short.

Definition 125 : Personally identifying information (PII) / personal data from GDPR

> *" 'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person;" [Art. 4 (1) GDPR]*

2.16. Definition of consent

In the area of personal data, e.g., the GDPR, the notion of consent is defined as follows:

> " 'consent' of the data subject means any freely given, specific, informed and unambiguous indication of the data subject's wishes by which he or she, by a statement or by a clear affirmative action, signifies agreement to the processing of personal data relating to him or her; " [Art. 4 (11) GDPR]

This is also in line with technical definitions, i.e., ISO 29100 defines consent as "personally identifiable information (PII) principal's freely given, specific and informed agreement to the processing of their PII" [250]. Alike definitions can be found in many legal texts, e.g. in Germany in § 630d BGB[91]. Further, there is the notion of 'informed consent' from medicine[92] with the idea to inform a patient previous to medical treatments, e.g., medical consent forms [505]. Note, in German the BDSG explicitly requires the data subject's consent to be in written form[93].

This thesis needed a general notion of consent, hence the following definition has been generalised from the one in the GDPR. The references to the data subject and personal data have been removed. It was further broadened to include legal persons[94]. However, note that this definition, without the further additions found in the original legal definitions, might not conform to BDSG or GDPR.

Definition 33 : Consent

> *A natural or legal person's consent means any freely given, specific, informed and unambiguous indication of the natural or legal person's wishes by which the natural or legal person, by a statement or by a clear affirmative action, signifies agreement to a specified action relating to the natural or legal person.*

[91] § 630d BGB states "Vor Durchführung einer medizinischen Maßnahme [...] ist der Behandelnde verpflichtet, die Einwilligung des Patienten einzuholen." [§ 630d (1) BGB].

[92] following *Beauchamp*: "[t]he practice of obtaining informed consent has its history in, and gains its meaning from, medicine and biomedical research." [30, p. 1].

[93] "Die Einwilligung bedarf der Schriftform [...]" [§ 4a (1) BDSG].

[94] It is not of interest for this work and not in scope to identify how and to what extent legal persons can make statements of consent, please refer to [335] or [20].

3 — New definitions and harmonised terms and notation for malleable signatures

Overview of Chapter 3

Chapter 3 previews many new terms and definitions that have been contributions of this thesis, introducing along the notations used to describe them. For an increased accessibility terms and definitions are already presented in brief in this chapter, even if the thorough analysis of an existing body of work is presented later in the thesis. A link to the respective chapter or section, where the context and details of those definitions are given, is provided. Chapter 3 shall provide the reader with a solid understanding of what a malleable signature scheme is, the integrity and confidentiality protection they offer, and become familiar with the terms and notation of this thesis.

3.1. Detailed notion of integrity (see Chapter 6)

A major contribution of this thesis is an extended notion of integrity that on the one hand captures the legal requirements and on the other hand allows to describe malleable signatures and classic digital signatures in a technically precise manner. A document protected by a malleable signature can be modified during a subsequent authorized sanitization or redaction and the signature verifies as valid on the modified document. In contrast, the definition of an EF-CMA secure digital signature scheme (see Sec. 5.3.4) defines that nobody other than the signer can produce a valid signature for a modified document.

The resulting Definition 113, stated in the following, was obtained from an analysis of the treatment of authorized modifications in existing technical definitions (see Chapter 5) and in legal requirements set forth in legal codes and texts (see Chapter 4). The details of this extended notion of integrity are given in Chapter 6.

Definition 113 : Data integrity

> *Data integrity is a specific state of data that is verifiable (potentially by a third party).*
> *A verification of data integrity that yields a **valid** output corroborates that the integrity-protected data has **not** been modified (or destroyed) in an unauthorized manner **since***
> *the integrity protection mechanism has been applied to the data.*
> *A verification of data integrity that yields an **invalid** outcome corroborates that the integrity-protected data has been modified (or destroyed) in an unauthorized manner **after***
> *the integrity protection mechanism has been applied to the data.*
>
> *Notes for Definition 113:*
>
> *Note 1 Definition 113 does not exclude authorized modifications.*
>
> *Note 2 Definition 113 requires to precisely capture in an integrity policy (e.g., ISO 10181-6 [340]) what constitutes an unauthorized or authorized change for this data. Compliance with this integrity policy is implicitly assumed to be checked to the full extent by the integrity verification mechanism.*

Note 3 *Definition 113 sees data integrity in accordance with ISO 10181-6 as "a specific invariant on data" [340]. Following this, the integrity protection mechanism "detects the violation of internal consistency" [340]. "A datum is internally consistent if and only if all modifications of this item satisfy the relevant integrity security policies." [340]*

Note 4 *Definition 113 is not concerned with the external consistency (e.g., as in [125]), or veracity [215] of the data.*

Note 5 *Definition 113 is hinting that integrity should allow the detection of integrity violations by third parties. However, it is generic in this respect and thus the level of detection (as defined in Sec. 6.3) may vary, and it depends on the technical detection achieved by the mechanism.*

Note 6 *Definition 113 explicitly leaves it open to define if the current state of integrity can be checked by a third party or not with an additional property.*

3.2. Definition of blocks ($m[i]$) and related terms

Malleable signatures in this thesis are "[...] built upon the paradigm of dividing a message into submessages" [509]. Malleable signatures are used to "[...] sign a document [...] that is partitioned into t blocks, for some constant t." [12]. Following, not only *Ateniese et al.*'s statement from [12], this thesis assumes the compartmentalisation of a message m into what this thesis calls blocks. In order to harmonise and to describe malleable signatures consistently, this section formalises the concept behind blocks and partial messages by means of several definitions: functions to decompose a message into blocks, functions to compose a message from blocks, 'contains' relation between a message and a message's block, and notions of partial and complete messages.

Definition 34 : Block

Using the decomposition function $\mathsf{decompose}()$*, the message*[95] m *is split into* ℓ *blocks, for some constant* ℓ*, with each block* $m[i] \in \{0,1\}^*$.

Notes for Definition 34:

Note 1 $|m|$ *denotes the number of blocks, i.e.,* $|m| = \ell$.

Note 2 *This thesis makes no assumption what is contained in a block, e.g., it is not assuming that each block captures a single semantic concept, nor that all blocks of* m *are of equal length.*

Note 3 *This thesis assumes that the blocks are not allowed to overlap.*

Note 4 *The maximum number of blocks in a given message is limited*[96] *by the number of the smallest unit this message can technically be decomposed into, i.e., a single bit* $\{0,1\}$.

Note 5 *Blocks are important for the integrity policy of malleable signatures: A single block is the data that can be subjected to a modification operation by a* Sanitizer *(introduced in Sec. 3.6.1). The* Signer *can only designate complete blocks for admissible modification.*

The data structure of the document is usually considered when decomposing a document into blocks. Therefore, this thesis will facilitate tree-, sequence- or set-like notions for messages and their blocks as needed, e.g., for the sequence $m[j]$ is the j^{th} element of m, and $(\{m[j]\}, \{e_{1,j}, e_{2,j}\})$ in a tree means that the node $m[j]$ has two edges, one to the node $m[1]$ and one to the node $m[2]$. This notation allows the thesis to make further statements about the structure of blocks.

[95] 'message' can be used interchangeably with 'data' or 'document' in this context

[96] because as per Note 3 blocks do not overlap

Definition 35 : Set of Blocks

If the order of blocks is of no importance for the information contained in the blocks of m, then this thesis uses a mathematical set-like notation: $m = \{m[1], \ldots, m[\ell]\}$ where $\ell \in \mathbb{N}^+$.

Note 1 Definition 35 implies that m contains only pairwise distinct blocks: $\forall 1 \leq i \leq \ell$ and $1 \leq j \leq \ell : \neg\exists m[i] = m[j]$ with $i \neq j$ and $m[i], m[j] \in m$.

Definition 36 : Sequence/List or one-dimensional Array of Blocks

If the sequence of blocks matters for the information contained in the blocks of m, then this thesis uses an array-like indexing scheme to denote the exact positions within a sequence of blocks. The index starts at 1. This yields the following representation: $m = (m[1] \,||\, \ldots \,||\, m[\ell])$ where $\ell \in \mathbb{N}^+$. Where "$||$" denotes a uniquely reversible concatenation of the blocks.

Note 1 Definition 36 allows a sequence m of length ℓ to contain two (or more) blocks which are equal: $m[i] \in m$ and $m[j] \in m$ and $m[i] = m[j]$ where $i \neq j$ and $i, j \in \{1, \ldots, \ell\}$.

Definition 37 : Tree of Blocks

When the blocks form the nodes of a tree, the thesis uses the following tree notation: A tree T is a tuple (V, E). V is the set of nodes containing the blocks: $V = \{n_1, \ldots, n_\ell]\}$ with the content of each node (c_n) being the message content, i.e. $c_i = m[i]$ for all $i \in \{1, \ldots, \ell\}$. $E \subseteq V \times V$ is a set describing all edges between the nodes: $E = \{e_{i,j}|$ when there exists an edge between $n[i]$ and $n[j]\}$.
Additionally, the notation is as follows:

1. *There exists a root node without a parent, denoted as r.*
2. *A node will be denoted as n_i.*
3. *Every node, except the root-node, has exactly one parent.*
4. *Every node, including the root-node, can have an arbitrary number of children.*
5. *A node without children is called a leaf(-node).*
6. *Every node can contain data, the data of node n is denoted as c_n.*
7. *The nodes are not ordered.*

Following next are definitions describing the initial decomposition into blocks and relations between messages and their blocks.

Definition 38 : Decomposition of Message into Block(s) and Composition of Message from Block(s)

Regardless of the structure of m, this thesis assumes there exists a decomposition function $\mathsf{decompose}()$ that splits m into ℓ block(s), such that the composition function $\mathsf{compose}()$ taking the same ℓ block(s) as input reconstructs them into a message that is equal to m. The two algorithms $\mathsf{decompose}$ and $\mathsf{compose}$ are PPT efficient. They are assumed to work correctly, such that:

$$\forall m : m \leftarrow \mathsf{compose}(\mathsf{decompose}(m)) \,.$$

Definition 39 : Block Contained in a Message

Regardless of the structure, this thesis denotes by $m[j] \in m$ that the block $m[j]$ is a block from the decomposed message m.
To ease the readability, this can be denoted for several blocks as follows:

$\{m[b], \ldots, m[d]\} \in m$ denotes that all blocks that are listed are contained in m, i.e., $m[b]$, $m[c]$, and $m[d]$.

Definition 40 : Removal of a Block from a Message

> *Regardless of the structure, $m \setminus m[j]$ denotes the removal of block $m[j]$ from m. To ease readability, this can be denoted for several blocks as follows:*
>
> *$m \setminus \{m[c], \ldots, m[e]\}$ denotes the removal of all blocks that are listed from m, i.e., $m[c]$, $m[d]$ and $m[e]$.*

To define the terms *partial* and *complete message*, the above concept of blocks can be applied:

Definition 41 : Complete Message

> *Regardless of the structure of m, let $\{m[1], \ldots, m[\ell]\} \in \mathcal{S}$ denote the blocks of m.*
>
> *A message m' with the blocks $m'[1], \ldots, m'[\ell']$ is complete with respect to the above m, if all the blocks of m' are pairwise equal to the blocks of m:*
>
> $$|m|' = |m| = \ell = \ell',$$
>
> *and*
>
> $$\forall i \leq \ell' : \exists j \in \{1, \ldots, \ell\} \text{ such that } m'[i] = m[j],$$
>
> *and*
>
> $$\forall j \leq \ell : \exists i \in \{1, \ldots, \ell'\} \text{ such that } m[j] = m'[i].$$

This thesis defined the complete message as the decomposed original message. A message is said to be *partial* if it is different in at least one block, and hence the notion of a partial message is defined as follows:

Definition 42 : Partial Message

> *Regardless of the structure of m, let $m[1], \ldots, m[\ell]$ denote the blocks of m. A message m' with the blocks $m'[1], \ldots, m'[\ell']$ is partial with respect to the above m, if there exists at least one block in which the messages differ and at least one block in which they are equal:*
>
> $$\exists i \leq \ell' \text{ such that } m'[i] = m[j], \text{for all } j \in \{1, \ldots, \ell\},$$
>
> *and*
>
> $$\exists a \leq \ell' \text{ such that } m'[a] \neq m[b], \text{for all } b \in \{1, \ldots, \ell\}.$$

In line with the above Definition 42 of a partial message and the correctness of the decomposition function a complete message can always be composed from all blocks, leading to the formulation of Definition 43.

Definition 43 : From all Blocks the Complete Message can always be Composed

> *Let $\mathcal{S} \leftarrow \mathsf{decompose}(m)$ be all the blocks of message m after correct decomposition. Given $\mathcal{S}$ a correct $\mathsf{compose}$ algorithm always reconstructs m without the need for any additional information, by $m \leftarrow \mathsf{compose}(\mathcal{S})$.*

When one block is missing, Definition 43 shall not hold. Namely, if $\mathcal{S}'$ describes all the blocks that are available in m', and if by Definition 42 there is at least one block $m[a]$ missing, $m[a] \in m$ but $m[a] \notin m'$, then under the assumption of the correctness of $\mathsf{decompose}$ and $\mathsf{compose}$: $m' \leftarrow \mathsf{compose}(\mathcal{S}')$, but $m \not\leftarrow \mathsf{compose}(\mathcal{S}')$. Due to $m[a] \notin m'$, at least one block of m was not supplied to $\mathsf{compose}$, hence m' will be a partial message with respect to m. This leads to Definition 44, which follows from correctness and by Definition 42.

Definition 44 : Without Additional Information a Complete Message cannot be Composed from any of its Partial Messages

> *Let $\mathcal{S} \leftarrow \mathsf{decompose}(m)$ be all the blocks of message m after correct decomposition. Let m, decomposed into $\mathcal{S}$, be the complete message and $\mathcal{S}'$ be the blocks of a partial message with respect to above m. Then without additional information the complete message m can never be composed by $\mathsf{compose}$ from the blocks of the partial message $\mathcal{S}'$, because:*
>
> $$\exists a \text{ such that } \exists m[a] \in \mathcal{S} \text{ and } m[a] \notin \mathcal{S}'.$$
>
> *Thus, $m[a]$ from the complete message is not known and thus m cannot be constructed from $\mathcal{S}'$.*

3.3. Definition of modification (see Sec. 5.5.2)

For this thesis what constitutes a modification of data depends on the system which processes the data. Henceforth, a changed document falls into exactly one of three classes: unmodified, modified with authorization and modified without authorization. The following definitions capture the relation between unauthorized modifications, most often described in technical integrity definitions, and the notion of authorized modifications used for this thesis. For this relation this thesis defined a modification in general in Definition 104. Unauthorized as well as authorized modifications are a special case of a modification in general. These definitions are the result of a thorough analysis of technical (Chapter 5) and legal definitions (Chapter 4) of data integrity; see Sec. 5.5.1 and Sec. 5.5.2 for more background.

3.3.1. Definition of unauthorized modification (see Sec. 5.5.2.1)

An unwanted modification, inflicted on data by a third party with a malicious intent or due to error is defined for this thesis as follows:

Definition 103 : Unauthorized Modification

An ***unauthorized*** *modification of data is any alteration of data (including the creation or insertion or deletion of data) that results in* ***unauthorized*** *observable consequence(s) during further processing when compared to the further processing of unmodified data (including, but not limited to an observably different unauthorized output).*

Notes for Definition 103:

Note 1 Definition 103 defines an unauthorized modification as application-dependent, by its reference to further processing.

Note 2 Definition 103 makes no statement about the fitness or suitability for the further processing — neither of the original data nor of the modified data.

Note 3 Definition 103 makes a detection of the occurrence of a modification possible due to the notion of observable consequence(s).

Note 4 Definition 103 includes deletion of data as a modification. Depending on the definition deletion is also an attack on availability.

Note 5 Definition 103 does not differentiate why the modification has occurred (i.e. it makes no distinction between a malicious active attacker or an error).

3.3.2. Definition of authorized modification (see Sec. 5.5.2.3)

Definition 105 : Authorized Modification

An ***authorized*** *modification of data is any alteration of data (including the creation, insertion or deletion of data) that results* ***only*** *in* ***authorized*** *observable consequences during further processing when compared to the further processing of unmodified data (including, but not limited to an observably different* ***and authorized*** *output).*

Notes for Definition 105:

Note 1 Definition 105 defines an ***authorized*** *modification as application-dependent, by its reference to the further processing.*

Note 2 Definition 105 makes no statement about the fitness or suitability for the further processing of the original data nor of the modified data.

Note 3 Definition 105 makes a detection of the occurrence of an ***authorized*** *modification possible due to the notion of observable consequences.*

Note 4 Definition 105 includes deletion of data as an ***authorized*** *modification. Depending on the definition of availability the action of deletion is an attack on availability.*

Note 5 Definition 105 does not differentiate why the modification has occurred (i.e., it makes no distinction between a malicious active attacker, ***an authorized third party*** *or an error).*

3.4. Definition of authorized modification and the notions 'malleable', 'redactable' and 'sanitizable'

This section provides an overview of the related work considered influential for the terminology in the greater context of malleable signature schemes. Within the scope if this thesis, two forms of authorized modifications will be differentiated: redactions and sanitizations. Fig. 16 highlights the differences between the output after those modification operations, and indicates the intended result of the signature verification if signed by a malleable signature schemes allowing operations on the string 'SECRET'.

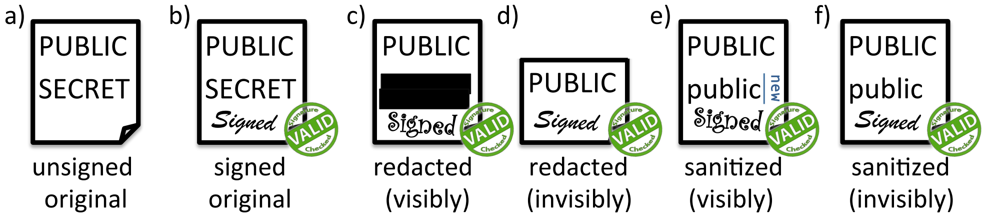


Figure 16. Results after different types of modification, if the modification of 'SECRET' is authorized, then the signature still verifies as valid: (a) original, (b) signed original, (c) visibly redacted + visible position of redaction, (d) invisibly redacted, (e) visibly sanitized + visible position of sanitization, (f) invisibly sanitized

3.4.1. Notion 'malleable'

In this thesis the term 'malleable' is used to describe a relaxation of the unforgeability requirement of classic digital signature schemes. In more detail, malleability allows the Signer to define an authorized modification, such that this modification will not fall in the class of forgeries forbidden by the unforgeability notion. The term 'malleable' was used in a similar way by *Boneh, Segev, and Waters* [59] and also by *Wei, Coull, and Reiter* [501] in 2011, by *Ahn, Boneh, Camenisch, Hohenberger, abhi shelat, and Waters* [6] in 2012, and by *Chase, Kohlweiss, Lysyanskaya, and Meiklejohn* [119] in 2014.

However, this thesis would like to explicitly state that the notion 'malleable' is not following the meaning by *Chik-How* [123]: Contrary to a modifiable signed content they define a "[...] security consideration [that] is called a malleable signature scheme. It means that if an adversary A used the known signature on a message to produce a different valid signature on the same message without knowing the private key." [123].

In this thesis malleability of signed documents is understood to be controlled. Controlled means that the malleable signature scheme MUST allow to restrict the modification if desired. This notion of malleability was termed "Targeted Malleability" [59] in 2011 by *Boneh, Segev, and Waters* and introduced formally in the realm of homomorphic encryption:

> "We introduce a precise framework, [...] ensuring that the malleability of a scheme is targeted only at a specific set of "allowable" functions." [59]

The above notion was given only for the context of homomorphic encryption in 2011; in 2012 the work on homomorphic signatures by *Ahn, Boneh, Camenisch, Hohenberger, abhi shelat, and Waters* defines "the notion of slightly homomorphic signatures, or P-homomorphic signatures" [6] as follows:

> "With such [P-homomorphic] signatures, it is possible for a third party to derive a signature on the object m' from a signature of m as long as $P(m, m') = 1$ for some predicate P which captures the "authenticatable relationship" between m' and m." [6][97]

[97] "object" [6] is understood here to be synonymous to document or message

The above definition by *Ahn et al.* is able to use the *predicate* P to capture existing definitions including: "arithmetic, homomorphic, quotable, redactable, transitive signatures" [6].

In 2014 the notion of "malleable signatures" was introduced by *Chase, Kohlweiss, Lysyanskaya, and Meiklejohn*. It is also general and captures "transformations (such as redactable [...], sanitizable [...], quotable [...], and transitive signatures" [119]. *Chase et al.* define a malleable signature as follows:

> "A signature scheme is malleable — alternatively, homomorphic — if, given a signature σ on a message m, one can efficiently derive a signature σ' on a message $m' = T(m)$ for an allowed transformation T." [119]

The authors further explain that the above informally given definition (formally given in [119]) is even more general than the definition by *Ahn et al.* and note that "[...] the definition of Ahn et al. requires that an adversary cannot generate a signature on a message m' unless there exists some previously signed message m and an allowed transformation T such that $T(m) = m'$." [119] A "key notational difference" [119] in the definition by *Chase et al.* compared to *Ahn et al.* is that it does not consider the malleability to be described as a predicate on input and output messages but as *transformations*. As noted by the authors, this "allows [...] to meaningfully capture transformations like exponentiation [...]" [119]. The above makes a difference in generality and it makes a difference for properties like unforgeability. However, this thesis would like to note the following:

The Signer defines the allowed malleability. Even if the more recent notions of malleable signatures — namely the one by *Ahn et al.* [6] as well as the one by *Chase et al.* [119] — are different to the ones before they both require the allowed malleability to be defined by the Signer. Both have the verification of the signature validating that only authorized subsequent modifications, defined by a predicate [6] or a set of transformations [119], have taken place. Unforgeability, a required basic security property, for malleable signatures must capture that subsequent modifications done without the knowledge of the required secrets get detected as unauthorized and flagged as forgeries.

Following the research by *Chase et al.* [119] and *Ahn et al.* [6], this thesis considers malleable signatures as a broader umbrella notion for signatures that allow subsequent modifications to signed documents. In line with *Boneh et al.* [59] the modifications are defined. Hence, this thesis defines malleable signature as follows:

Definition 45 : Malleable Signature Scheme (High-Level)

A malleable signature scheme is a secure digital signature scheme that

- *allows the Signer to authorize one (or more) semi-trusted entities, called Sanitizer, to carry out designated modifications to certain designated blocks of a document and produce a valid signature of the resulting authorized modified document;*
- *allows a Sanitizer to produce the valid signature on the authorized modified document, which can be produced without interaction with the original Signer;*
- *prohibits even a designated Sanitizer, let alone a third party, to produce a valid signature for any unauthorized modification;*
- *hence the Sanitizer must only be semi-trusted.*

The Sanitizer is semi-trusted as the Sanitizer's subsequent modifications are limited to those authorized. Due to the control the signing party delegates only selected subsequent modifications on selected blocks of the content of the signed document to the Sanitizer. Hence, the delegatee only receives the right to produce the "appropriate alteration" [347] bearing a signature which still verifies and still relates to the signing party. Thus, the delegatee does not receive the full rights, only pre-defined subsequent modifications are allowed. Hence the notion of the Sanitizer being semi-trusted.

In the following, this thesis will look at two forms of malleable signatures in detail: *sanitizable signature schemes* (SSS) and *redactable signature schemes* (RSS). Each of them has been discussed previously in the literature and they can be separated by the different modifications that can be authorized, i.e., they have different malleability.

3.4.2. Notion 'redaction'

The malleability inherent in the redactable signature schemes (RSS) discussed in this thesis has been introduced in [273, 455] in 2002. Both works describe a signature scheme that allows removing parts from signed data without this removal invalidating the signature. The concept was termed as "content extraction signature" [455] by *Steinfeld, Bull, and Zheng*, while *Johnson, Molnar, Song, and Wagner* introduced the same concept a little bit differently and termed it "homomorphic signature" [273].

> "A CES [Content Extraction Signature] allows the owner, Bob, of a document signed by Alice, to produce an 'extracted signature' on selected extracted portions of the original document, which can be verified to originate from Alice by any third party Cathy, while **hiding** the unextracted (removed) document portions."[98] [455]

The above describes that the action of "hiding the unextracted (removed) document portions" [455] would keep a verifiable valid signature. In the remainder of the thesis the high-level understanding of redactable signatures's functionality and properties is based on the work of *Steinfeld, Bull, and Zheng* [455].

In 2003, in an extended version [457] of [455] *Steinfeld et al.* acknowledge that *Johnson et al.* independently introduced a closely related functionality under the term 'redactable signature' "[...] although the privacy security notion is not formalized [...]"[99] [457]. The notion is defined by *Johnson et al.* [273] as follows:

> "Redactable signatures are intended to model a situation where a censor can delete certain substrings of a signed document without destroying the ability of the recipient to verify the integrity of the resulting (redacted) document." [273]

This thesis' understanding of an RSS follows the functionality and intent described by *Johnson et al.* and *Steinfeld et al.* [273, 455]. The following definition already takes the legal requirements for confidentiality protection into account (see discussion in Sec. 8.1). This thesis uses the term 'redaction' as used and coined in [273]:

Definition 46 : Valid Redaction

> *In general, a redaction denotes the act of deleting, i.e., removing, one or more blocks of a given message. The redacted block(s) are not reconstructable from the redacted message.*
>
> *In the context of a redactable signature scheme:*
> *A **valid redaction** is a redaction of a signed message that results in a still valid signature over the redacted message.*
>
> *Notes for Definition 46:*
>
> *Note 1 As blocks, following the definition given in this thesis, are not limited to only message content but also include structural information, e.g., tree of blocks, Definition 46 is not limited to allowing only the deletion of content, but includes also the deletion of structural information.*
>
> *Note 2 Definition 46 does not require the deletion to be undetectable. This thesis will use the special symbol ■ as a substitution of the previous information to denote that a redaction has taken place. If the redaction is detectable, then there can be different levels of detection, e.g., the length of the content, or the position of the redacted blocks relative to the rest of the message.*
>
> *Note 3 Definition 46, in general, does not require the message to be signed. In the context of a redactable signature scheme, a valid redaction is a redaction of a signed message that is authorized by the signer and thus results in a still valid signature over the redacted message.*
>
> *Note 4 Definition 46 explicitly states that a redacted block is removed without the possibility to reconstruct it from the modified message, i.e., its confidentiality is protected as it is irreversibly removed/deleted/hidden.*

[98] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.
[99] The American English spelling from original work is retained for terms and quotes.

Henceforth, the malleability of an RSS is limited to only the deletion of pre-determined admissible blocks including other admissible information. Other admissible information could be the structure (see Sec. 12.5.2 on page 305 or $struct\mathcal{RSS}$ in Sec. 14.9 on page 399) or the admissibility itself (see Sec. 14.5.1.1 on page 379). Redactions are depicted in Fig. 16 (c) and (d).

3.4.3. Notion 'sanitization'

The malleability of sanitizable signature schemes (SSS) in this thesis follows the description of *Ateniese et al.* from 2005 [12]:

> "We define a sanitizable signature scheme as a secure digital signature scheme that allows a semi-trusted censor to modify certain designated portions of the message and produce a valid signature of the resulting (legitimately modified) message with no interaction with the original signer." [12]

A little more detailed, but in the same direction, is the later description by *Brzuska et al.* from 2009 [64]:

> "Sanitizable signature schemes [...] allow a signer to delegate signature rights in a controlled way. Namely, the signer can determine parts of the message which a designated party, the sanitizer, can later modify but such that the authenticity and integrity of the remaining parts is still guaranteed. In particular, even the sanitizer should not be able to change inadmissible parts of the message and produce a valid signature for such illegitimate transformations." [64]

This thesis' understanding of SSS follows the functionality and intent described by *Ateniese, Chou, de Medeiros, and Tsudik* in [12]. It also already takes the legal requirements for confidentiality protection into account (see discussion in Sec. 8.1).

Definition 47 : Valid Sanitization

> *In general, a sanitization denotes the act of modifying one or more blocks of a given message. The original block that was modified is not reconstructable from the sanitized message.*
>
> *In the context of a sanitizable signature scheme:*
> *A* ***valid sanitization*** *is a sanitization of a signed message that results in a still valid signature over the sanitized message.*
>
> *Notes for Definition 47:*
>
> *Note 1 This thesis' and Definition 47's notion of modification includes performing the identity transformation, i.e., no action on the actual contents of a block, which will be denoted as 'touching a part'.*
>
> *Note 2 Definition 47 does not require the modification to be undetectable. If the sanitization is detectable, then there can be different levels of detection, e.g., the position of the sanitized blocks relative to the rest of the message.*
>
> *Note 3 Definition 47 does not require the message to be signed in general, however then a sanitization is just another name for a modification. When used in the context of a sanitizable signature scheme, a valid sanitization is a sanitization of a signed message that is authorized by the signer and thus a valid sanitization results in a still valid signature over the sanitized message.*
>
> *Note 4 Definition 47 explicitly states that a sanitized block is overwritten during the modification without the possibility to reconstruct it from the sanitized message, i.e., its confidentiality is protected as it is irreversibly replaced / overwritten / modified.*

Henceforth, the malleability of sanitizable signature schemes allows admissible blocks to be altered to arbitrary strings. Sanitizations are depicted in Fig. 16(e) and (f).

A sanitization that actively does a sanitization, but does not modify the message is said to be 'touching' the message. The notion of Sanitization given above and in Definition 47 includes actively performing no action on a block. This is different than not applying the sanitization on a block at all. Therefore, this active sanitization will be called 'touching' the block. Still, this active action leaves the block's contents 'as-is'. For example by making a backup of a block's content, then deleting the block, and then re-inserting the previously generated exact backup of it at the exact same location in m. Touching a block results in m' being parsed exactly as m would have been, as it does not change the message's structure or contents. However, depending on the actual cryptographic scheme, the active sanitization, even if it does not modify the message, might influence a message's malleable signature. Henceforth, touching a block is considered as a possible sanitization action.

3.4.4. Separate security models of RSS and SSS (see Chapter 15)

Sanitizable and redactable signatures are similar in their functionality but have cryptographically different security models: *De Meer, Pöhls, Posegga, and Samelin* have shown and proved in [133] that even though they seem to be related due to similarities in functionality, the cryptographic aims differ. Their security models substantially differ on a more detailed level [133] (see Appendix A publication n^{o} 19):

- no black-box transformation can transform an RSS into an SSS, and
- a black-box transformation of an SSS even with additionally strengthened security only yields an RSS with weaker security properties than the state of the art. [133]

These findings — obtained during the course of the thesis and published in [133] — are an additional reason for the split focus (see Sec. 1.4) of this thesis: RSS and SSS are improved separately (see Chapter 13 for SSS and Chapter 14 for RSS). A focussed discussion of the results are given in Chapter 15 and of course in the paper [133].

3.5. Authorized modifications policy (ADM), immutable ($m[\mathsf{fix}]$) or admissible blocks ($m[\mathsf{ADM}]$), instructions for modifications (MOD), and all admissible messages ($\mathsf{span}_{\vdash}(m, \mathsf{ADM})$)

The Analysis Result 8 obtained in this thesis yields that an authorized modification needs to be stated in a policy. The definition of authorized modification can be found in Sec. 5.5.2.3 of this thesis; it defines what is authorized to be modified through the policy. This admissible modification policy is denoted as ADM and defined generically as follows:

Definition 48 : Admissible Modification Policy (ADM)

> *The admissible change policy denotes the subsequent modifications that are authorized by the Signer, as well as the party authorized to do them.*

This very broad definition captures the possibilities for authorization and authorized principals as discussed in Sec. 3.7. The following definitions interpret it as a function that codifies the policy.

Definition 49 : Description of an Admissible Message with respect to an Admissible Modification Policy ($\mathsf{ADM}(m)$)

> *ADM is a function, that, on input of a given message m returns a decision $d \in \{0,1\}$, denoting if the sanitization of m is admissible with respect to the admissible modification policy ADM. In case of an error it will return $\perp$.*

The admissibility of a single block with index notation $m[i]$ is denoted by $\mathsf{ADM}(i)$ and defined as follows:

Definition 50 : Description of a Single Admissible Block with respect to an Admissible Modification Policy ($\mathsf{ADM}_m(i)$ or $\mathsf{ADM}(i)$)

> *ADM denotes those blocks that are authorized for modification. ADM_m is a function, that, for a given message m, takes as input an identifier i that identifies a block of m and returns a decision $d \in \{0,1\}$, denoting if the sanitization of the block $m[i] \in m$ is admissible ($d = 1$) or not ($d = 0$) with respect to the admissible modification policy ADM. In case of an error it will return $\perp$.*

If the message m is clear from the context this thesis abbreviates this notation to just $\mathsf{ADM}(i) = \{0,1\}$. Note, what is used as an identifier depends on the structure, e.g., i is the index for a list.

EXAMPLE

Examples for the use of notation: If $\mathsf{ADM}_m(5)$ returns 1, then the fifth block of m may be sanitized. An example for a representation of ADM could be a bitmap $\{0,1\}^\ell$ for a message with ℓ blocks. A value of '1' at position i in this bitmap denotes that the i-th block of the message, denoted as $m[i]$, is admissible for modification, a '0' at position i that $m[i]$ is not admissible. It could also be represented as a list of indices for the admissible blocks and the total number of blocks, e.g., $\mathsf{ADM} = (\{1,3,5,7,9\},10)$ means in a message with 10 indexed blocks those with odd indices are authorized for subsequent modification.

The following notation is used to refer to **all** blocks of a message m which are admissible:

Definition 51 : Description of All Admissible Blocks with respect to an Admissible Modification Policy ($m[\mathsf{ADM}]$)

$m[\mathsf{ADM}]$ is a set that contains those blocks of a message m that are authorized to be modified with respect to the admissible modification policy ADM:

$$m[\mathsf{ADM}] = \{m[i] \mid m[i] \in m \text{ and } \mathsf{ADM}_m(i) = 1\}\,.$$

The above corresponds to $\mathsf{ADM}_m(i) = 1$ and implies the following:

$$\mathsf{ADM}_m(i) = 0 \Rightarrow\; m[i] \notin m[\mathsf{ADM}]\,,$$

$$\mathsf{ADM}_m(i) = 1 \Rightarrow\; m[i] \in m[\mathsf{ADM}]\,.$$

Further, this thesis will need to refer to all those blocks that are not admissible for modification. They are called immutable blocks and are defined as follows:

Definition 52 : Description of All Immutable Blocks ($m[\mathsf{fix}]$) with respect to an Admissible Modification Policy

$m[\mathsf{fix}]$ is a set that contains those blocks of a message m that must not be modified with respect to the admissible modification policy ADM:

$$m[\mathsf{fix}] = \{m[i] \mid m[i] \in m \text{ and } \mathsf{ADM}_m(i) = 0\}\,.$$

The above corresponds to $\mathsf{ADM}_m(i) = 0$ as $\forall m[\mathsf{fix}][i] \in m[\mathsf{fix}] : \mathsf{ADM}_m(i) = 0$. This implies the following relations:

$$\mathsf{ADM}_m(i) = 1 \Rightarrow\; m[i] \notin m[\mathsf{fix}]\,,$$

$$\mathsf{ADM}_m(i) = 0 \Rightarrow\; m[i] \in m[\mathsf{fix}]\,.$$

When thinking about m being a consecutive sequence of bits, then from the above notation follows that this thesis assumes that the overlapping of admissible and fixed blocks is not allowed. There may however be applications that allow those overlaps by an additional encoding of the actual message into the cryptographic message space, e.g., the mapping of an XML node to a block by means of an identifier like the XML id which is described in Sec. 18.1.

Finally, it was explicitly discussed in [294] that it is helpful to denote if a given policy matches a message. This would be annotated as follows:

Definition 53 : Correct Admissible Modification Policy with respect to a Message ($\mathsf{ADM} \subseteq m$)

Using a symbol used in mathematical notation of sets, this thesis denotes with $\mathsf{ADM} \subseteq m$ if the admissible modification policy ADM fits the given message m. If $\mathsf{ADM} \not\subseteq m$, then the admissible modification policy does not correspond to the message and to prevent unspecified behaviour $\mathsf{ADM}(m)$ should return $\perp$.

To denote what shall be done during a modification, this thesis will use the abstract container MOD. It is defined as follows:

Definition 54 : Modification Instruction (MOD)

MOD is a container that describes all the modifications that an application wants to apply to a message or its block(s), i.e., sanitizations or redactions.

To ease notation this thesis will also use MOD to denote the function that modifies the input message m into the modified m' according to the description, i.e., the function is the description of subsequent modifications. The notation is as follows:

Definition 55 : Apply Modification Instruction to Message ($\mathsf{MOD}(m)$)

$\mathsf{MOD}(m)$ denotes the application of the modification instruction MOD to modify the input message m into the modified m':

$$m' \leftarrow \mathsf{MOD}(m)\,.$$

To refer to the detailed modification inflicted on an individual message's block the notation MOD also applies to one single block $m[i]$. This thesis uses the following notation:

Definition 56 : Apply Modification Instruction to Message's Block ($\mathsf{MOD}(m_{NEW}, i)$)

$\mathsf{MOD}(m)$ denotes the application of the modification instruction MOD to modify one single block $m[i]$ of input message gets modified into $m'[i]$:

$$m'[i] \leftarrow \mathsf{MOD}(m_{NEW}, i)\,.$$

Note, m_{NEW} denotes the new information for the block with the identifier i.
Note, the message m is assumed to be clear from context of the modification instruction.

This thesis will further use "for all $i \in \mathsf{MOD}$" to indicate a loop iterating and retrieving all identifiers for blocks for which modifications are defined in a given modification instruction MOD.

Definition 57 : Empty Modification Instruction ($\mathsf{MOD} = \emptyset$)

An empty modification instruction is denoted by $\mathsf{MOD} = \emptyset$. If MOD is empty and is applied to a message, the message is still said to be sanitized. On an empty MOD the function that modifies the input message m will return the same message as output, i.e., $m' = m$. This thesis will refer to this as 'touching' a message.

The same applies if an empty modification instruction is applied to a single block, i.e., $\mathsf{MOD}(m[i]_{NEW}, i)$ where $m[i]_{NEW} = m[i]$. This thesis will refer to this as 'touching' a block.

The action of 'touching' is inherent to the action of modification and stated explicitly in MOD: 'Touching' or 'no operation' is a modification. Hence, MOD will explicitly state that this action is to be carried out. Examples to implement this behaviour are:

- the identity function is applied as the operation that derives the contents of the new block, e.g., $m_{NEW}[i] := m[i]$, then $m'[i] \leftarrow \mathsf{MOD}(m_{NEW}, i)$;
- special symbol to indicate that no operation on content shall be performed, e.g., $m'[i] \leftarrow \mathsf{MOD}(\mathsf{NOP}, i)$.

As a result touching the block addressable via i means that the modification instruction for an $m[i] \in$ ADM does not change the string of bits this message's block is comprised of:

$$\text{for all } i \in \mathsf{MOD} : m'[i] \leftarrow \mathsf{MOD}(\mathsf{NOP}, i) \text{ results in } m' = m\,.$$

Of course, depending on the scheme touching could result in leaving a 'virtual fingerprint' in the signature, resulting in a change in the accountability of the whole message or only of the touched block.

It is often needed to describe a set containing all possible modified messages or a subset with only those modified messages that are valid with respect to a given policy ADM.

Definition 58 : All Possible Modifications ($\{\mathsf{MOD}(m)\}$)

Using symbols used in mathematical notation of sets, MOD directly applied to a message will denote the set containing all possible modifications of m, including m itself and $\emptyset$. This is denoted as $\{\mathsf{MOD}(m)\}$.

This is helpful to keep notation compact. The additional brackets are used to indicate the set generating behaviour. For a given message m the set $\{\mathsf{MOD}(m)\}$ contains all possible derivable modifications. Note that $\{\mathsf{MOD}(m)\} \subseteq \mathcal{M}$, but $\{\mathsf{MOD}(m)\}$ can be as big as the whole message space ($\mathcal{M}$) for sanitizable schemes.

To describe only those modifications that are admissible, this thesis uses the following 'compacted' notation: $\mathsf{span}_{\vdash}(m, \mathsf{ADM})$, which is derived from *Chang et al.* [117].

Definition 59 : All Admissible Modifications ($\mathsf{span}_{\vdash}(m, \mathsf{ADM})$)

> *$\mathsf{span}_{\vdash}(m, \mathsf{ADM})$ is a set that contains all those messages that can be derived from m by modifications which are authorized by the modifications policy ADM.*

Some times additional brackets are used to indicate the set behaviour. The set $\{\mathsf{MOD}(m)\}$ with all possible modifications of the message m gets reduced to contain only those messages y as elements that are authorized by ADM to yield $\mathsf{span}_{\vdash}(m, \mathsf{ADM})$:

$$\{\mathsf{span}_{\vdash}(m, \mathsf{ADM})\} = \{y \mid y \in \{\mathsf{MOD}(m)\} \text{ and } \mathsf{ADM}(y) = 1\}\,.$$

The notation $\mathsf{span}_{\vdash}(m, \mathsf{ADM})$ is especially compact as it works for any structure of m. Note, the notation makes no statement on how to derive or if it is is efficient to derive $\mathsf{span}_{\vdash}(m, \mathsf{ADM})$ for any m and any ADM. Already, *Brzuska et al.* [64] pointed out that checking if $m^* \in \mathsf{span}_{\vdash}(m, \mathsf{ADM})$ holds "[...] may not be efficiently verifiable [...]" [64].

Finally, to state that all modification instructions contained in MOD are admissible with respect to ADM this thesis uses the following notation:

Definition 60 : Modification Instructions are Admissible Modifications ($\mathsf{MOD} \subseteq \mathsf{ADM}$)

> *Using symbols used in mathematical notation of sets, $\mathsf{MOD} \subseteq \mathsf{ADM}$ is used to denote that MOD only contains modification instructions which are admissible under ADM.*
> *The opposite, $\mathsf{MOD} \not\subseteq \mathsf{ADM}$ is used to denote that a modification in MOD breaches the policy in ADM.*

This also allows to state $\mathsf{span}_{\vdash}(m, \mathsf{ADM})$ using a set generating description as follows:

$$\{\mathsf{span}_{\vdash}(m, \mathsf{ADM})\} = \{\mathsf{MOD}^{\ddagger}(m) \mid \mathsf{MOD}^{\ddagger} \subseteq \mathsf{ADM}\}\,.$$

In the above $\mathsf{MOD}^{\ddagger} \subseteq \mathsf{ADM}$ checks that only admissible modification instructions are applied to m by $\mathsf{MOD}^{\ddagger}(m)$. The work *Brzuska et al.* [64] introduced, among others, the notation $\mathsf{ADM}(\mathsf{MOD}) = \{0, 1\}$ to state whether or not MOD is within the ADM. When using that notation one yields the following slightly different notation to state the set of all possible authorized modified messages: $\mathsf{span}_{\vdash}(m, \mathsf{ADM}) = \{\mathsf{MOD}(m)^{\ddagger} \mid \mathsf{MOD} \text{ with } \mathsf{ADM}(\mathsf{MOD}^{\ddagger}) = 1\}$.

3.6. Entities and their secrets in MSS

The concept of malleable signatures introduces an additional party, henceforth called the Sanitizer. In this section the different roles are being separated by their keys; moreover they are defined and their trust relations are stated.

3.6.1. Definition of roles for involved entities separated by their secrets

Figures 17 and 18 show the workflow: First, the generation of a signed document by signing a document with a sanitizable or a redactable signature scheme. Second, the Sanitizer and what it is authorized to sanitize gets identified. Third, the Verifier uses the signature to verify that no unauthorized modifications have been made to the document after the generation of the signature. The Verifier further uses the trusted public-key-to-identity binding to identify the origin of the signed document. This binding can be obtained through additional means, like public key certificates previously issued by a TTP.

This thesis identified in Definition 20 (see Sec. 2.10 on page 29) for CDSS that any entity interacting in the workflow of digital signatures will be in at least one role. As with CDSS, the roles in an RSS or SSS can be categorised by the need for knowledge of a certain secret, or by the absence of this need.

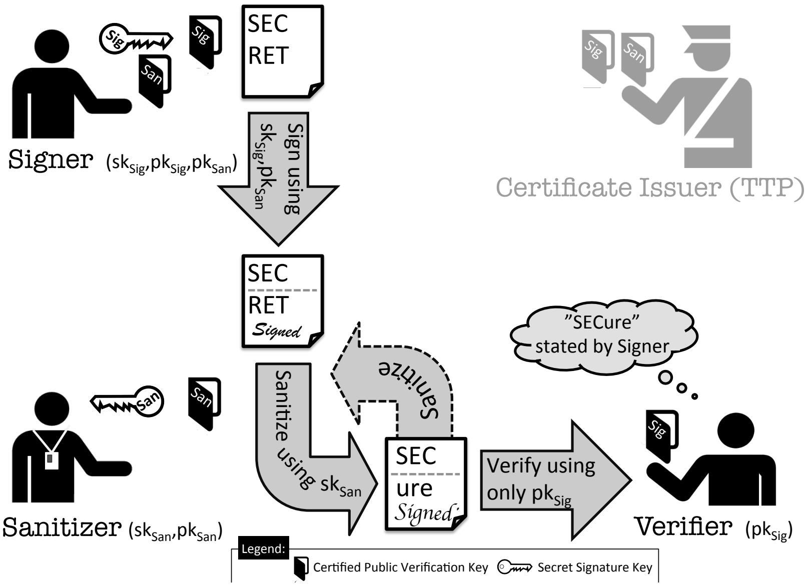

SEC
RET
Signer (sk$_{Sig}$,pk$_{Sig}$,pk$_{San}$)
Sign using sk$_{Sig}$,pk$_{San}$
Certificate Issuer (TTP)
SEC
RET
Signed
"SECure" stated by Signer
Sanitize
Sanitize using sk$_{San}$
SEC
ure
Signed'
Verify using only pk$_{Sig}$
Sanitizer (sk$_{San}$,pk$_{San}$)
Verifier (pk$_{Sig}$)
Legend:
Certified Public Verification Key
Secret Signature Key

Figure 17. Workflow to sanitize "SECRET" into "SECure"; Verification needs just Signer's public verification key; Certificates can be **previously** issued by a TTP; TTP is offline and not involved in sanitization

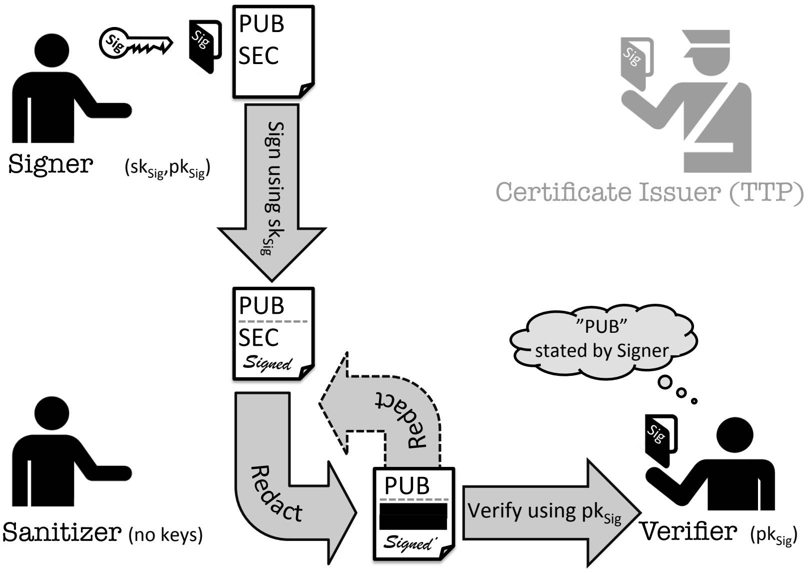


Figure 18. Workflow to redact "SEC"; Verification needs just Signer's public verification key; Certificate can be **previously** issued by a TTP; TTP is offline and not involved in redaction

The differences between the classic digital signature's roles described in Definition 20 and those in a MSS are highlighted using underlining:

Signer.

- Requires knowledge of the secret signing key $\mathsf{sk}_{\mathrm{sig}}$;
- does not know $\mathsf{sk}_{\mathrm{san}}$ or $\mathsf{sk}_{\mathrm{TTP}}$;
- generates signature σ using $\mathsf{sk}_{\mathrm{sig}}$, among others, as input to algorithm Sign;
- may use $\mathsf{sk}_{\mathrm{sig}}$ in algorithms other than Sign to generate other values than the signature, e.g., the proof π used by algorithm Judge.

Sanitizer.

- Requires knowledge of the secret sanitizing key sk_{san} (if needed);
- Does not know sk_{sig} or sk_{TTP};
- generates signature σ' using algorithm Sanit.

Verifier.

- Does not know sk_{sig}, sk_{san} or sk_{TTP};
- verifies signature σ or σ' using only public information, e.g., pk_{sig} , pk_{san} and pk_{TTP}.

Arbitrator.

- Does not know sk_{sig} or sk_{san} or sk_{TTP};
- executes the accountability resolution protocol, probably involving Signer or Sanitizer.

Certificate Issuer (as a trusted third party).

- requires knowledge of the secret certificate-issuing key sk_{TTP};
- does not know sk_{sig} and sk_{san};
- issues certificates binding an `id` string to the public key (e.g., knows pk_{sig} and pk_{san});
- trusted by all parties.

The different roles are defined as follows:

Definition 61 : Signer

> *This thesis uses Signer to denote the party in the role of Signer, which produced the initial signature over a message m using his secret signature key sk_{sig}. This is also the party authorized to define the admissible blocks by ADM and immutable blocks through $m[\mathsf{fix}]$. It further defines the party (or parties) that can assume the role of a Sanitizer. Depending on the signature scheme, but not required in general for MSS, this party uses additional public key(s) to restrict the party (or parties) that can assume the role of a Sanitizer, denoted as pk_{san}.*

Note on Terminology: Signer is the entity which initially signs. Although both the original signing as well as any following sanitization of a message produces a signature for that message and thereby could be 'signing' the message, this thesis uses the role Signer to refer only to the party that provided the initial first signature, all subsequent signatures are created by a party (or parties) in the role of a Sanitizer — assuming a secure signature scheme.

Definition 62 : Sanitizer

> *This thesis uses Sanitizer to denote a party in the role of Sanitizer, authorized by the party with the role Signer to modify (here used to mean either sanitize or redact) admissible blocks of a given signed message. Depending on the signature scheme, but not required in general for MSS, this party may require a secret sanitizer key sk_{san}.*

Note on Terminology: Sanitizer can sanitize or redact. In this thesis, to limit unnecessarily expanding the terminology, the role of a Sanitizer in a redactable scheme is limited to redactions, but will not be called redactor. However, note that RSS and SSS are different, though related concepts, which cannot be transformed into each other in a black-box transformation [133] (see Chapter 15 and the related publication [133] listed as n° 19 in Appendix A).

Definition 63 : Verifier

> *This thesis uses Verifier to denote the party in the role of Verifier, which seeks to verify a signature over a message m using only public information and preferably as little additional interaction with other parties as possible.*

Note on Terminology: Verifier is defined generally, but designated Verifiers are out of scope of the thesis. Definition 63 is general and would allow to include also the notion of designated verifiers, but this is out of scope (see Sec. 2.10.2.3). The definition does allow for requiring that the Verifier has to interact with other parties, before or during the process of signature verification. For example, a previous interactions is to obtain a trusted public key of another party.

3.6.2. Secrets and trust relations between entities in MSS

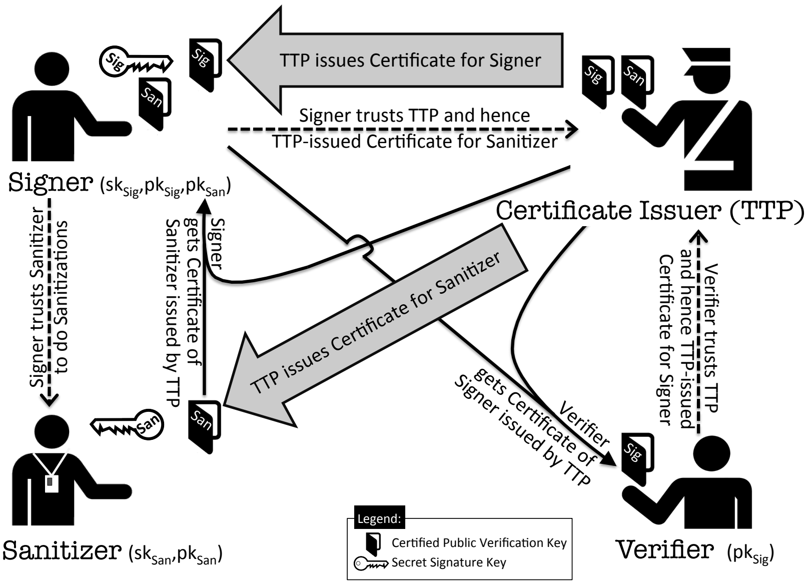

TTP issues Certificate for Signer
Sig
San
Signer trusts TTP and hence TTP-issued Certificate for Sanitizer
Signer $(sk_{Sig},pk_{Sig},pk_{San})$
Certificate Issuer (TTP)
Signer trusts Sanitizer to do Sanitizations
Signer gets Certificate of Sanitizer issued by TTP
TTP issues Certificate for Sanitizer
Verifier gets Certificate of Signer issued by TTP
Verifier trusts TTP and hence TTP-issued Certificate for Signer
Sanitizer (sk_{San},pk_{San})
Verifier (pk_{Sig})
Legend:
Certified Public Verification Key
Secret Signature Key

Figure 19. The four involved roles in a malleable signature and their trust relations (dashed lines); Note, public key certificates issued by a trusted certificate issuer are assumed to be previously distributed out of band

The Sanitizer is a designated fourth party introduced in the malleable signature scheme. As the signature shall retain verifiability of integrity and authenticity of origin — equally to the functionality in a CDSS — asymmetric keys can be used to build the necessary identifiability and also the needed trust relationships for the entity in the role of the Sanitizer. Fig. 19 shows the four parties and the trust between them. The Sanitizer will, if needed, get an asymmetric key pair, denoted as $(\mathsf{sk}_{\mathrm{san}}, \mathsf{pk}_{\mathrm{san}})$.

3.7. Authorization of principals and actions

In general, an authorization grants a principal an action on an object. The two strands of MSS looked at in this thesis differ from each other and from CDSS also in what they allow to authorize. The authorized actions on the signed document (the object) clearly differ:

- CDSS allow no action,
- RSS allow the removal / blackening, and
- SSS allow the action of substitution.

Moreover, the two forms of secure MSS differ in their ability to select the authorized principal(s)[100]:

- existing[101] RSS do not allow to select the authorized principal, as redact is a public operation.
- Sanitizer-accountable[102] SSS select the authorized principal by setting the public key of the Sanitizer.

[100] The CDSS allows to authorize no subsequent action, hence there is also no principal to become authorize.

[101] Note, the *pubacc*$\mathcal{RSS}$ scheme proposed in this thesis offers accountability for RSS and thus makes redaction non-public (see Sec. 14.1.4 and 14.13).

[102] Secure schemes are considered accountable, unforgeable and private. Accountability composed from Signer- and Sanitizer-accountability is the relevant property in this discussion.

This difference is also affecting the offerings that can be made by SSS and RSS for the property of signer control (see Sec. 7.2.1 on page 168). Fig. 20 depicts this as a matrix.

	no actor can be authorized	specific actor can be authorized
no specific action can be authorized	CDSS classic digital signature scheme	
specific action can be authorized	RSS redactable signature scheme	SSS sanitizable signature scheme

Figure 20. Differences of CDSS, existing secure SSS and RSS[101] in their ability to select authorized actions and authorized principals.

3.8. Adversary models, game-based property definitions and security assumptions

The adversary's only goal is to break a security property, which in the remainder of this thesis will be described using game-based definitions (see Sec. 3.8.1). In general, the adversary is considered as an additional maliciously acting principal, which in general can take any one of the roles described in Sec. 3.6.1. There are some general assumptions made about the adversary's general capabilities (see Subsections 3.8.2 and 3.8.3). Malleable signature schemes come with certain cryptographic security properties. A list of those properties in a harmonised presentation is provided in Chapter 11.

Existing work fixed the adversary's role to Verifier or Sanitizer. The adversary in the role of the Verifier has no knowledge of any of the secrets. This can be seen as an adversary in its weakest form. A slightly stronger adversary is a malicious Sanitizer. The adversary in the role of the Sanitizer in an SSS has knowledge of the Sanitizer's secrets.

Still, this adversary shall not be able to break the security property of immutability which confines the Sanitizer to only those subsequent modifications that were authorized by the Signer (see Definition 153 on page 261). When no adversary, neither in the role of the Verifier nor in the role of a Sanitizer, can break immutability, then this leaves the Signer with no argumentation to technically repudiate having generated the signature for data if the signature still verifies for the data under the Signer's verification key. This yields the Signer-accountability and non-repudiation capabilities of RSS and SSS.

Note, in the course of this thesis, a joint publication with *C. Brzuska* and *K. Samelin* [69] (see Appendix A publication n^{o} 17) extended the adversary model (see Sec. 13.3): It considers a malicious Signer as an adversary on privacy. Seeing the Signer as the adversary stems from the application domain of privacy: If a subsequent modification is done by a rightful Sanitizer to protect the data subject's privacy, then it shall not be possible that a malicious Signer allows the Verifier to undo the modification. Hence, this extension of the adversary model goes beyond the state of the art and it is especially interesting for privacy.

3.8.1. Games describing adversary's capabilities and the scheme's security property

The security properties in this thesis are defined and described as games, e.g., as done in [65, 66]. The main idea behind this rigorous way of describing the adversary's capability and the resistance a security property has to offer is as follows: To break a property, a theoretical adversary has to win the game while being bound to the rules set forth in the game. First, the game will generate fixed values, usually key pairs. In this step, the game might also draw some random coins. Second, during the course of the attack, a strong adversary can usually not only observe but also influence the generation of values; even those that are depending on secrets the adversary has no access to. An example is an adversary that can inject messages that then are signed by the Signer. To model this, the adversary can be given access to so-called *oracles*, e.g., a signing oracle which uses the Signer's secret to generate signatures for adversary generated messages. Oracles perform cryptographic operations according to the signature scheme's algorithm or the algorithm explained in the game on behalf, but not under the control of the adversary. From the adversary's point of view, the oracle performs as a closed 'black-box'. The input values only known to the oracle are specified as inputs for the oracle. The input for an algorithm that the adversary can influence and control is denoted as '$\cdot$' or as '$\cdots$' if the input spans more than one variable.

Finally, the adversary algorithm is denoted as $\mathcal{A}$ and has input values describing (a) which previously generated values it has access to, (b) which oracles it can use. It will produce an output which will have to obey certain requirements set forth in the properties description, to win the game.

EXAMPLE

Example: The following adversary algorithm $\mathcal{A}$ creates a message-signature pair $(m*, \sigma*)$:

$$(m*, \sigma*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk}_{\mathrm{sig}}, \cdots)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})\,.$$

The adversary algorithm $\mathcal{A}$ is supplied with $\mathsf{pk}_{\mathrm{sig}}$ and $\mathsf{pk}_{\mathrm{san}}$, i.e. it knows these two public keys. This still does not mean that the adversary can influence or generate them — the adversary just knows them. Especially, the adversary does not know the secret $\mathsf{sk}_{\mathrm{sig}}$ to perform the Sign algorithm. Instead, it can use an oracle denoted as $\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk}_{\mathrm{sig}}, \cdots)$. The security parameter 1^λ is also not chosen by the adversary, but known. The input values only known to the oracle are specified as direct inputs for the oracle, here the security parameter and the second input to Sign, the secret key $\mathsf{sk}_{\mathrm{sig}}$, is not known to the adversary; This is denoted as $(1^\lambda, \cdot, \mathsf{sk}_{\mathrm{sig}}, \cdots)$.

3.8.2. Security Assumption 1 (Secret keys are not known to the adversary)

Security Assumption 1:

> *Secret signature generation keys sk_{sig} or sk_{san} from the key pairs $(\mathsf{sk}_{sig}, \mathsf{pk}_{sig})$ and $(\mathsf{sk}_{san}, \mathsf{pk}_{san})$ are unknown to and protected against unauthorized use by an adversary unless the adversary is in a role with secret key access, e.g., a malicious $\mathsf{Sanitizer}$ knows sk_{san}.*

This thesis assumes that "[t]he key pair owner is the only entity that is authorized to use the private key to generate digital signatures. In order to prevent other entities from claiming to be the key pair owner and using the private key to generate fraudulent signatures, the private key must remain secret." [355].

Note, there are strong adversary models in which the adversary is in the role of a $\mathsf{Sanitizer}$ and thus has access to the $\mathsf{Sanitizer}$'s secret $\mathsf{sk}_{\mathrm{san}}$, e.g., [69] (see Sec. 13.3 for more details).

3.8.3. Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed)

Security Assumption 2:

> *All entities can assure that the public key they are using during the execution of a signature scheme's algorithm uniquely corresponds to the alleged entity.*

This requires a indirect or direct mutual trust of the entities in the $\mathsf{Certificate\ Issuer}$. This means that for accountability, the $\mathsf{Verifier}$ or $\mathsf{Arbitrator}$ is always[103] able to link a used public key $\mathsf{pk}_{\mathrm{san}}$ or $\mathsf{pk}_{\mathrm{sig}}$ to the correct identity. This includes the ability to obtain the public keys when needed during the execution of a signature scheme's algorithm. With above assumption all parties can be convinced that they can deduct trust in a public key from the $\mathsf{Certificate\ Issuer}$, as they are able to use it as a root of trust[104].

[103] Or at least with an overwhelming probability as sometimes stated in the cryptographic literature, e.g. [64].

[104] In a hierarchically organised infrastructure one public key must be initially trusted and becomes the "[...] root of trust, or "trust anchor" [...]" [1, p. 132–133].

The mechanisms for public key certificate issuing usually involve a trusted third party, but it could also be decentralised. However, the details of this are outside the scope of this thesis (see Sec. 1.4.3). Hence, the malleable signature schemes presented in this thesis will make no assumption of how public keys are linked to meaningful identities. They assume the Certificate Issuer is working correctly and that public key distribution is working. However, the schemes will need to support linking public keys to natural or legal persons.[105]

3.9. High-level definition of privacy for MSS (see Sec. 11.6.3 for SSS and Sec. 11.13.3 for RSS)

In this thesis the notion 'privacy' is related to a cryptographic property of MSS called privacy. Cryptographically, the standard privacy notion as introduced formally by *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk* [64] is already stronger than other definitions from MSS literature (see Sec. 12.1). The following definition of privacy is used in this thesis and discussed in more detail in Sec. 11.6.3:

Definition 155 : Standard Privacy

> *A* ***private*** *scheme prevents Verifiers from recovering any information (esp. the original value) about block(s) of m, which are no longer in the sanitized block(s) of m', through a valid signature σ' over m'.*
>
> *Note: Information leakage through the semantic content of the modified message m' itself, which is given to the adversary, is out of scope.*

Fulfilling the security property of standard privacy — following its above definition — inhibits an adversary to revert the modification done by a sanitizer. To give an example, from the redacted document in Fig. 8 on page 11 the adversary is not able to learn what information was there before it got blackened. It is the achievement of confidentiality through a privacy property that makes malleable signatures a valuable tool: it allows them the blackening/anonymization of personally identifying information to conform with data protection rules, or protect the confidentiality of trade secrets; while being able to share and distribute the remaining information from signed documents with verifiable integrity and authenticity. The state-of-the-art cryptographic privacy from *Brzuska et al.* presented at PKC '09 [64] used in this thesis is further discussed in detail in Sec. 11.6.3. Cryptographically it is strong enough to limit attacks that would allow leakage from looking at the derived message's signature and the derived message. The research conducted for this thesis, and works by others, show that several current schemes, claimed to provide such a leakage-freeness, are not private under *Brzuska et al.*'s definition (see Sec. 12.1). If a scheme is private according to *Brzuska et al.*, an adversary has no advantage over guessing when trying to identify which one of the two adversary-supplied structurally equal input messages was used to derive a valid sanitized / redacted message. Where structurally equal means that the two adversary supplied messages must not allow to leak the choice of message from immutable blocks or structure.

The following example is for the notion of privacy as given by *Brzuska et al.* [64]. It shall explain how the adversary game for privacy works on high-level. Alongside it introduces the general concept of adversary games as they are the used method for all formal cryptographic property definitions given in this thesis.

EXAMPLE

Example: The privacy property is described as a game for an attacker: The adversary can use information that is publicly available to identify the outcome of a random coin toss which was tossed in secret, i.e. it remains a 'black box' for the attacker. Fig. 21 visualises the privacy game. The game consists of the following steps:

1 Let $m^* = m^*[1] \,||\, m^*[2] = AAA \,||\, BB$ and $m^{**} = m^{**}[1] \,||\, m^{**}[2] = AAA||DDDD$ be the adversary supplied message where block 1 is fixed and block 2 can be modified.

2 The adversary supplies two modification instructions (denoted as MOD^* and as MOD^{**}) that can be applied to m^* or m^{**} respectively.

3a Assume that for a redactable scheme the modification instructions would be
to redact both second blocks $m^*[2]$ and $m^{**}[2]$. Thus, the result is $m' = AAA$.

[105] See Requirement 5.

3b Assume that for a sanitizable scheme the modification instructions would be to modify each messages' second block to the same string: $m^*[2]' = m^{**}[2]' = CCC$. Thus, this results in $m' = AAA \,||\, CCC$.

4 Iff[106] applying MOD* to m^* and MOD** to m^{**} results in the same m', they are considered structurally equal inputs.

5 The unobservable 'Black-box' is now choosing at random either m^* or m^{**} and signs and sanitizes it to output m' to the adversary.

6 The adversary has to state which input was used, i.e. m^* or m^{**}.

The adversary wins iff it correctly identifies the used input from just m',σ' with non-negligible probability, i.e., is better than randomly guessing.

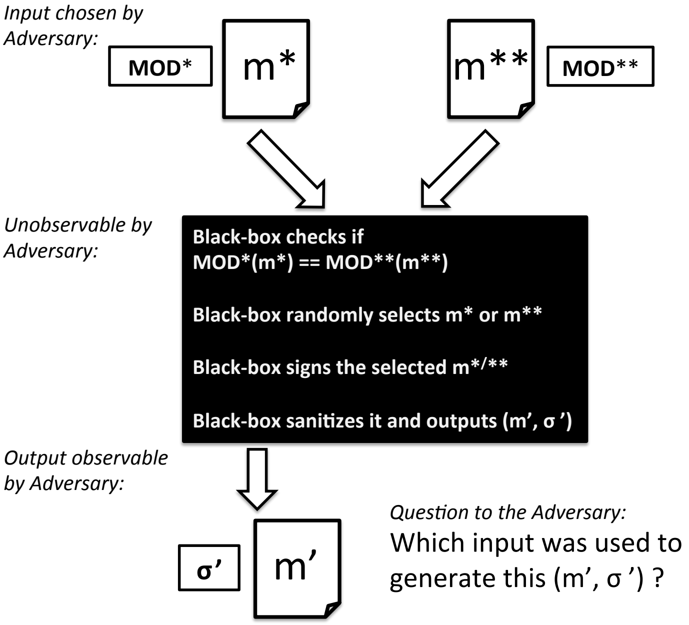

Input chosen by Adversary:
MOD*
m*
m**
MOD**
Unobservable by Adversary:
Black-box checks if MOD*(m*) == MOD**(m**)
Black-box randomly selects m* or m**
Black-box signs the selected m*/**
Black-box sanitizes it and outputs (m', σ ')
Output observable by Adversary:
σ'
m'
Question to the Adversary:
Which input was used to generate this (m', σ ') ?

Figure 21. High-level view of the privacy game (following *Brzuska et al.* [64]): The adversary should not be better than guessing when deciding which of the adversary provided input message produced the black-box's output.

This notion of cryptographic privacy is important in the scope of the thesis: It is this capability that protects the confidentiality of personal data or trade secrets. This thesis contains a more fine-grained discussion of the existing definitions of it in Sec. 11.6.3 for SSS and in Sec. 11.13.3 for RSS. Beyond the state of the art this thesis offers an even stronger cryptographic privacy notion, namely the previously mentioned newly defined *strengthened privacy* (see Sec. 15.1.3).

3.10. High-level definition of accountability for MSS (see Chapter 6, Sec. 11.6.6, or Sec. 13.1)

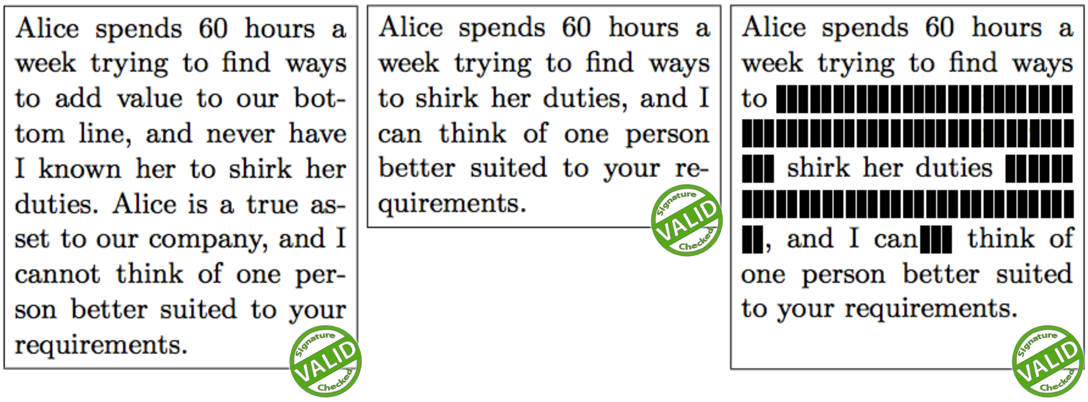
Related to the question what probative value is appointed to a document, the question of who is accountable for a document — in its current form — becomes important. In general, the Verifier of a malleable signed document could differentiate — by verification of the accompanying signature — four general states of the document regarding its modifications:

- **Unknown**: An error occurred rendering the document's state unidentifiable (e.g., due to a signature that was technically detected, but could not be parsed and understood correctly by the software/application/user).

[106] 'Iff' means the mathematical conjunction, i.e., 'if and only if'.

- **Invalidly Modified**: The signed document has been subject to modification(s) not foreseen and not authorized by the signer.
- **Validly Modified**: The signed document has been subject to only modification(s) which were foreseen and authorized by the signer.
- **Unmodified**: The signed document has not been subject to any subsequent modification.

However, a Verifier might not always be able to differentiate between all states, this is also shown in Fig. 23. Recall the overview of different forms where Fig. 16 d and Fig. 16 f showed authorized modifications that were transparent, i.e. invisible. *Johnson et al.* [273] offer a helpful illustration about the difference between visible and invisible occurred modifications. This is depicted in Fig. 22. They note that "the meaning of the original document has been violated" [273] if a text like the one in the middle is crafted. They point out that "[t]he right shows the corresponding redaction obtained if deletions are disclosed explicitly; note how the attempted trickery is revealed by this countermeasure." [273]



Alice spends 60 hours a week trying to find ways to add value to our bottom line, and never have I known her to shirk her duties. Alice is a true asset to our company, and I cannot think of one person better suited to your requirements.

Alice spends 60 hours a week trying to find ways to shirk her duties, and I can think of one person better suited to your requirements.

Alice spends 60 hours a week trying to find ways to ▮▮▮ shirk her duties ▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮▮, and I can▮▮▮ think of one person better suited to your requirements.

Figure 22. Transparent vs. visible redactions: Left is **original**, signed with an RSS where one character equals one redactable block, i.e., each character can be redacted but the order of characters must be preserved; Middle is after a **transparent (i.e, invisible)** redaction; Right is after a **detectable** redaction where detection leaks the position (and thus the length) of each redacted blocks, i.e., character, symbolised by the special symbol ▮.
Example from *Johnson et al.* [273], slightly adapted.

The middle text depicted in Fig. 22 hides the information that redactions took place. This is a strengthened notion over privacy, as it goes one step further and also keeps the fact that redactions have happened somewhere confidential. This property has been termed illustratively by the term "invisibility", meaning that an RSS allows for "invisibly deleting (redacting) information that should be kept confidential even after the document has been signed" [230][107]. In the existing body of literature ,this term is also found under the notion of cryptographic transparency, e.g. by *Brzuska et al.* [64] or *Ateniese et al.* [12]. This property is discussed next.

3.10.1. High-level definition of cryptographic transparency for MSS (see Chapter 6, Sec. 11.6.4, or Sec. 11.13.4)

In a transparent scheme, a cryptographic security property described in detail in Sec. 11.6.4 and Sec. 11.13.4, a document that was modified with authorization and an unmodified document cannot be differentiated by a Verifier. An early definition, also naming this property as such, was given by *Ateniese et al.* [12]. This got partly formalised by *Brzuska et al.* [64]. By definition, a transparent malleable signature scheme does allow the Verifier alone, but only in the interaction with the Signer, to differentiate between those states. Fig. 23a depicts the differentiable states in a scheme where a Verifier can only differentiate between a modified or an unmodified signed document. It might be that the

[107] The Hitachi online glossary [230] is used as a source here because the term "invisibility" summarises and motivates the functionality fittingly and because *Kunihiko Miyazaki*, who is the author and co-author of many RSS papers [222, 344, 346–348], was at Hitachi [343].

scheme allowed authorized modifications, however they stay transparent to the Verifier's own algorithms. Fig. 23b depicts the differentiable states in a scheme where a Verifier can additionally differentiate occurred modifications, without additional interactions, into those that were authorized by the signer and an unmodified document. This thesis termed this property *public accountability*.

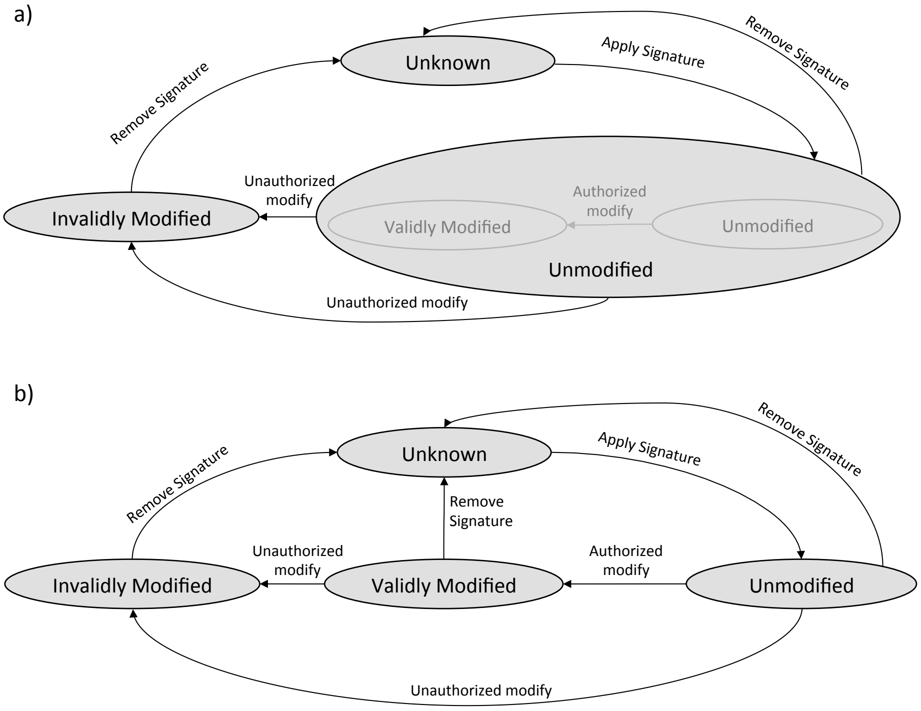


Figure 23. Different states that a Verifier can differentiate between in (a) transparent or (b) publicly accountable malleable signature scheme

The cryptographic property of transparency is of interest of this thesis as it prohibits the Verifier to detect subsequent modifications on its own, which is a contradiction to what integrity protection a classical digital signature would offer. This is a main point of this thesis' cryptographic research and requirement analysis. Without going into the details of the cryptographically notion of transparency here, a high-level view of this cryptographic property as it is of special interest for this thesis.

Definition 157 : Cryptographic Transparency

> ***Transparency*** *prevents third parties which have only access to a given valid message-signature pair (m,σ) and public information (e.g., pk_{sig} or pk_{san}) to decide which party (Signer or Sanitizer) is accountable for a given valid message-signature pair (m,σ).*
>
> *Notes for Definition 157:*
>
> *Note 1 Information leakage through the message itself is out of scope of Definition 157.*
>
> *Note 2 Transparency is cryptographically stronger than the cryptographic privacy notion (see Definition 155), i.e., transparency implies privacy [64].*
>
> *Note 3 Third party refers here to any party that is neither Signer or Sanitizer, i.e. an entity in the role of the Verifier. The third party has no access to non-public information; especially not to the secret keys (sk_{sig} or sk_{san}) or output generated involving them, i.e. no access to the proof π for the given valid message-signature pair (m,σ).*

3.10.2. Difference between the cryptographic properties transparency and public accountability for MSS (see Sec. 13.1)

Cryptographically, transparency is indeed a stronger privacy notion and, hence,

$$\text{transparency} \Rightarrow \text{privacy [64]}\,.$$

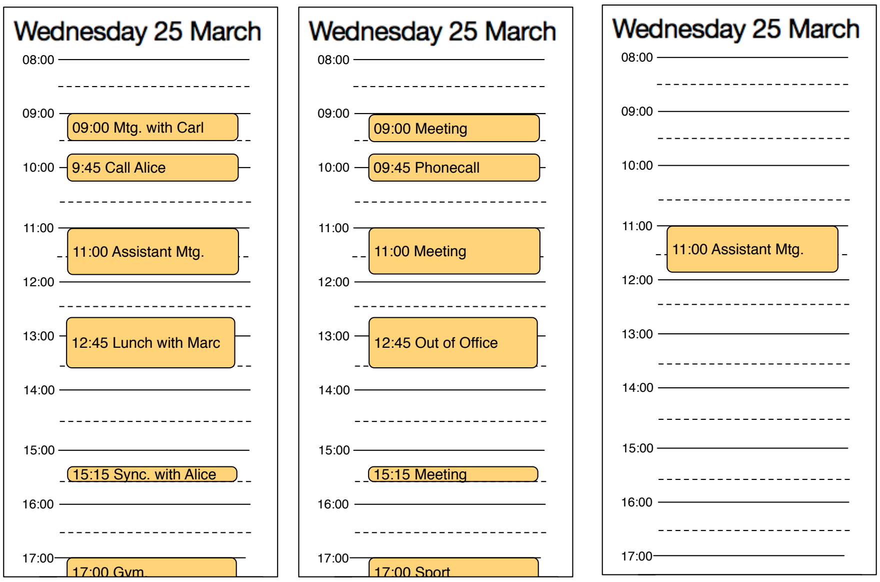
However, transparency inhibits the Verifier — considered a third party — from learning if there was an authorized change or not without any interaction. This thesis will include the discussion of detectability of authorized modifications in its extended integrity notion (see Chapter 6). Detectability is related to the property of accountability — it is the public form of accountability —, which is important for any legal and practical application.

3.10.2.1. Example for the usefulness of transparency

From the high-level descriptions and examples offered so far, one might argue that the detection of an occurred subsequent modification is obvious to derive by just observing that the output which still verifies was modified; the examples have shown handwritten changes (like in Fig. 6 on page 10) or redaction of printed supply chain documents (like in Fig. 8 on page 11) which leak obvious clues about the subsequent modifications. The examples shown so far also did not need the protection of cryptographic transparency, as subsequent modifications where either inherently wanted or not uncommon and detectable once an adversary had some knowledge other than just the signature, e.g. semantic information.

EXAMPLE

Example: Consider the example of a shared personal calendar, as depicted in Fig. 24.



(a) (b) (c)

Figure 24. Example of transparent modifications to calendar entries: (a) original, (b) transparently sanitized descriptions, (c) transparently redacted complete entries

Assume the calendar is signed to protect its authenticity of origin. Further assume that each entry in the calendar could have been marked as malleable in certain fields or in its entirety. While the left shows the full details, the middle or right views might be more appropriate to share with others. The middle version ((Fig. 24 (b)) was created by transparent sanitization of each entries' title to reveal a little more about why the scheduled person is busy at each time slot, but not the full information. The right-most version shows the calendar being redacted leaving only one entry containing the exact same information as in original. The latter is useful to distribute full information but only selected entries.

In the above example, the calendar's information-reduced views in Fig. 24 (b) and (c) do not obviously reveal the information loss due to the sanitization (Fig. 24 (b)) or the redaction (Fig. 24 (c)). The example shows that tailoring a view by redaction or sanitization, which reveals only as little as needed, can benefit from that operation being transparent: If the redaction is transparent, then a Verifier would not be able to reveal from the adjusted signature that an authorized subsequent modification took place.

3.10.2.2. Separation of transparency and non-interactive public accountability

To achieve detectability it is **insufficient** to negate the property of transparency: Assume the Verifier is reliably able to correctly detect $\frac{3}{4}$ of all Sanitizer involvements. Such a scheme is neither accountable and thus has no detectability because in $\frac{1}{4}$ of the cases the accountability is not correctly attributed, which is considered too unreliably to make it a security property. Neither is such a scheme transparent as the observer is indeed better than guessing when trying to detect Sanitizer involvements ($\frac{3}{4} \geq \frac{1}{2}$). Moreover, the sheer absence of transparency does not guarantee security against malicious a Signer/Sanitizer. Hence, this thesis describes a mechanism to allow a third party to decide whether a Sanitizer's secret key was involved in the signature generation or if only the signer's secret key was involved even in the presence of a malicious Signer/Sanitizer. The key's involvement can then be linked to the corresponding entities and determine the origin of the signature.

Definition 107 : Accountability

> ***Accountability*** *allows a predefined set of entities to determine a valid signature's origin (Signer or Sanitizer) according to a protocol, i.e., decide which party is accountable (split in Signer-accountability and Sanitizer-accountability) for the signature of a given valid message-signature pair* (m, σ).

Existing accountability properties in MSS literature allowed the Verifier to identify the accountable entity, e.g., the Signer could rebut the accusation of being accountable if at least one block of the document has been subsequently altered. However, the Verifier might need to involve the Signer or Sanitizer (or at least their respective secrets) and run a protocol in order to finally identify who has generated a signature for a valid message signature pair. This thesis notes that currently used and legally accepted digital signature schemes offer a more public and especially a non-interactive form of accountability: When a digital signature verifies accountability, it successfully identifies the involvement of a secret key that corresponds to the public verification key used; it does so without interaction and only involving trusted publicly available information. This thesis will henceforth refer to this accountability as *non-interactive public* accountability. The accountability property was discussed for SSS in Sec. 13.1 on page 310 and the proposed definition is as follows:

Definition 179 : Non-Interactive Public Accountability (PUB)

> *A sanitizable signature scheme satisfies* ***non-interactive public accountability*** *(PUB), if and only if for a valid message-signature pair* (m, σ)*, a third party can correctly decide whether* (m, σ) *originates from the Signer or from the Sanitizer without requiring an interaction with the Signer, the Sanitizer or any other party with access to secret keys, i.e. just from using public knowledge* $(m, \sigma, pk_{sig}, pk_{san})$.

Hence, transparency and non-interactive public accountability are exclusive, as depicted in Fig. 75.

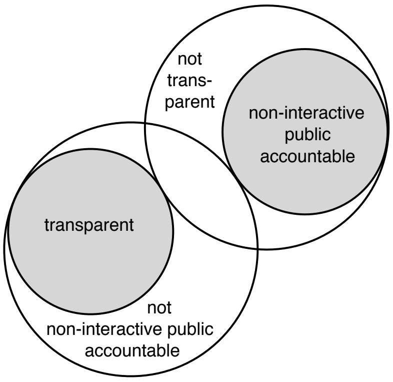


Figure 75. Cryptographic property of transparency in contrast to the new property of non-interactive public accountability
(same Figure as on page 314)

This relation is discussed in Sec. 13.1.3 in more detail when the new cryptographic property is defined in Sec. 13.1.

3.10.2.3. Challenge to remove the cryptographic property of transparency while upholding the property of privacy

Note that the property of transparency prohibits any form of non-interactive public accountability. One goal of this thesis is the removal of transparency to gain functional equivalence in this respect with exiting legally compliant signature schemes in order to award an increased probative value. Further, bear in mind that the cryptographic property of transparency is stronger than privacy, i.e., transparency implies privacy [64]. For the applicability of MSS for confidentiality protection of information it is of paramount importance to achieve a strong cryptographic privacy. Thus, it was this thesis' cryptographic challenge to provably rule out transparency to guarantee non-interactive public accountability without impeding privacy simultaneously.

This thesis presents the solutions to this challenge. It provides cryptographic constructions for redactable or sanitizable signature schemes that are secure with respect to the cryptographic properties of privacy and unforgeability, while they still allow non-interactive public accountability. Examples hereof are the newly constructed schemes $pubacc\mathcal{SSS}$ (see Sec. 13.4 on page 329), $blockacc\mathcal{SSS}$ (see Sec. 13.6 on page 340), and $pubacc\mathcal{RSS}$ (see Sec. 14.13 on page 417).

PART I

Proposed extended integrity definition & technical requirements for a high probative value

And the real worth of public key cryptography may not be fully realized in the real world until we look more closely at every way in which digital signatures differ from inked or stamped signatures, and public key encryption differs from sealed envelopes.

— G.R. Blakley
Twenty years of cryptography in the open literature,
IEEE Symposium on Security and Privacy, 1999 [53]

Contents

Chapter 4 High probative value and statutory evidentiary presumption for electronic documents

Chapter 5 Analysis of existing technical integrity and authenticity definitions with respect to authorized modifications

Chapter 6 Proposed extended integrity definition

Chapter 7 Technical requirements for integrity and authenticity protection to gain a high probative value

4 — High probative value and statutory evidentiary presumption for electronic documents

Overview of Chapter 4

The text of legal acts can be formulated to be technology-neutral allowing them to stay applicable even in case of technological advances. Especially, information technology (IT) legal acts are in need of definitions that are also technically sound and precise. Deviations between the technical and the legal terms may lead to wrong technical decisions when regulations' requirements are used as guidelines to design technical systems. Chapter 4 provides the background to understand what a signature scheme for electronic documents must achieve in order to give the signed electronic documents an increased probative value that is assigned by means of statutory rules of evidence. The assignment by statutory rules then gives a statutory evidentiary presumption. This means that the electronic document's contents contain a statement from the legal signatory which will be regarded as evidence with a legal presumption of authenticity.

Following the methodology explained in Sec. 1.5 this chapter gives the legal background and presents the 'interpretation' of the legal concepts used in the course of this thesis to derive the technical requirements for a high probative value for signed documents given in Chapter 7 to conclude Part I. This work follows *Roßnagel* [410] and the analysis only attributes a high probative value if all legal requirements and pre-requisites are fulfilled, i.e., the requirements must all be taken into account even if they are presented in different laws and legal texts' meaning are analysed together taking their correlation and interaction into account[108][410].

An overview of the selected legal texts taken into consideration in this Chapter is presented in Fig. 26. The German Code of Civil Procedure (ZPO a.F.[109] and ZPO n.F.) is used in this thesis as a precise and specific background to capture the concept behind *probative value*. The ZPO, among others, discusses an electronic document's probative value. For example, also ArbGG[110], VwGO[111], StPO[112] and FGO[113] following ZPO a.F. stated that documents "may be submitted in electronic form if they bear a qualified electronic signature in accordance with the Digital Signatures Act documents" [§ 41a para. 1 StPO a.F.]. With Regulation 910/2014 being in force, the texts have been changed following eIDASDG and no longer contain a reference to the German digital signature act (SigG); see Sec. 4.5.4.

[108] In German *Roßnagel* [410] states "Dabei geht die Prüfung davon aus, dass Voraussetzungen für, Anforderungen an und Rechtsfolgen von qualifizierten Signaturen aufeinander bezogen sind und in diesem Bezug aufeinander verstanden und interpretiert werden müssen, auch wenn die Regelungen in unterschiedlichen Gesetzen zu finden sind."[410].

[109] This thesis notes that this German legal text has just recently (18th of July 2017) been updated [99], but an earlier version [96] got analysed as well because older discussions in the legal debate are based on the former version.

[110] See for example § 46c para. 1 ArbGG a.F.

[111] See for example § 55a para. 1 VwGO a.F.

[112] See for example § 41a para. 1 StPO a.F.

[113] See for example § 52a para. 1 FGO a.F.

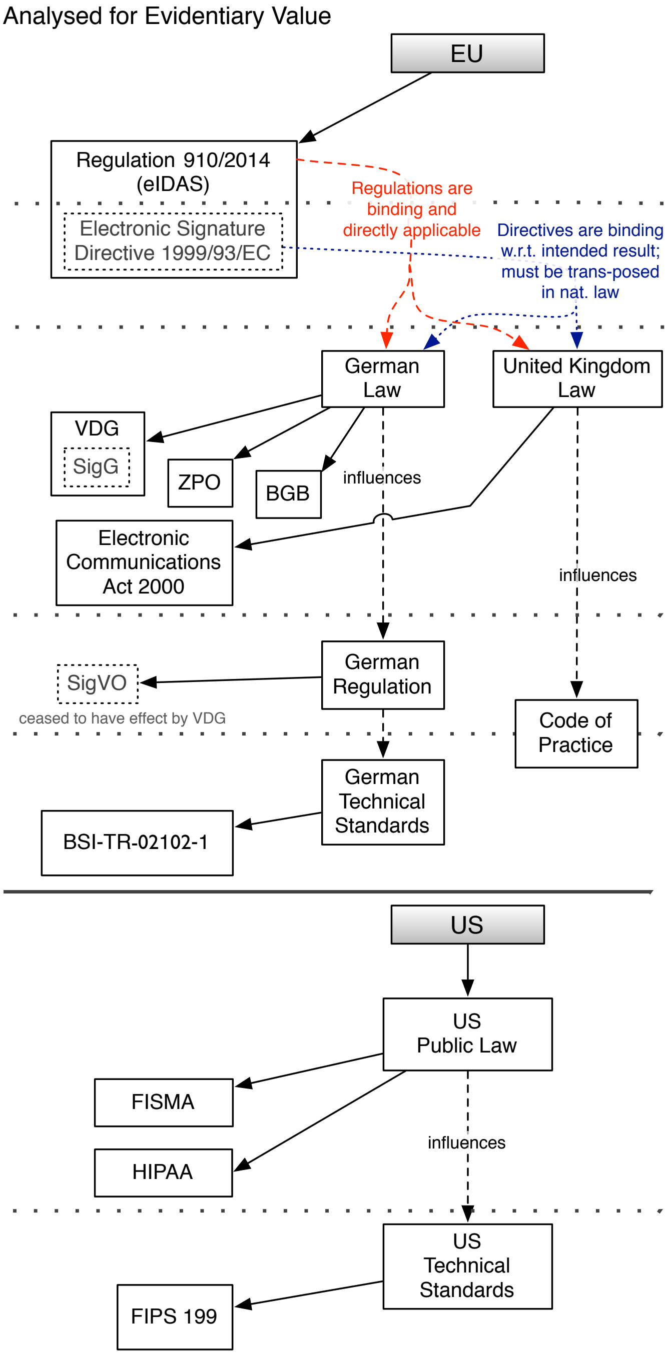


Figure 26. Overview of legal texts analysed in Chapter 4; dashed arrows show influences

ZPO was selected as main reference for simplicity, but without loss of generality as the other German laws do not disagree with respect to electronic documents and the assessment of the probative value for documentary evidence[114] and evidence taken by visual inspection[115].

[114] Handling of documentary evidence is also discussed for example in § 96 VwGO, §§ 249-256 StPO, and § 81 FGO.

[115] See §§ 86-93 StPO.

A look into the law of EU member states is also necessary as the EU legal texts do not explicitly state rules of evidence or the probative value for electronic documents signed with electronic signatures.[116] German legal texts offer legislative rules on handling electronic documents that bear an electronic signature as evidence. Further, German legal texts allow to assign the probative value as *statutory evidentiary presumption* using statutory rules of evidence for electronically signed documents. To identify that the focus on the EU and German legal texts indeed does not add a major bias, this chapter will also briefly discuss aspects of UK legislation regarding the probative value in Sec. 4.2.3 and US legislation regarding the term Data Integrity in Sec. 4.3.4. To provide a good readability for the non-legal readers and in order to stay self-contained, this thesis includes full text citations from concrete legal acts rather than just pointing to them.

4.1. Background on probative value and statutory evidentiary presumption from German Code of Civil Procedure (ZPO)

In 1998 the European Commission conducted a study on the digital signatures in law and practice [172, p. 127]. This study acknowledged that in 1998, which was prior to the Directive 1999/93/EC, "the legal value of digitally signed electronic documents is non existent" [172, p. 127]. The study also stated "that evidence laws can present barriers" [172, p. 104]. Even after Directive 1999/93/EC tried to harmonise member state legislation in 1999 and after Regulation 910/2014, there are still differences in how evidence laws work in different legal systems. Hence, this section discusses the legal recognition and the probative efficacy of electronic documents on the basis of concrete EU member state legislative acts.

Two legal notions are of interest for this thesis:

- Probative value[117], and
- rules[118] that assign statutory evidentiary presumption[119].

Both above notions are described in detail in this section on the basis of the German Code of Civil Procedure (ZPO) [96, 99][120], which offers detailed legislative rules on handling electronic documents that bear an electronic signature as evidence.

Generally, during the examination of evidence[121], the court establishes all the pieces of evidence presented to refute or proof that a certain fact holds or a certain action has taken place. The German Code of Civil Procedure (ZPO) describes general rules about the process for acquiring evidence in § 284 ZPO[122]:

> "The taking of evidence and the order for separate proceedings to take evidence, which is issued by a court order for evidence to be taken, are governed by the stipulations of Titles 5 through 11. The court may take evidence, provided it has obtained the consent of the parties to do so, in the manner it deems suitable. [...]"[123] [§ 284 ZPO]

Individual rules might apply from the applicable legislative acts that govern the individual topic of the case, but as these require a precisely instantiated legal case they are not further discussed in this thesis.

[116] Regulation 910/2014 makes general statements like "[...] admissible as evidence [...]" [Article 25, para. 1 Regulation 910/2014] or "[...] equivalent legal effect as handwritten signatures [...]" [Article 25 para. 2 Regulation 910/2014]. However, only for qualified electronic seals in Article 35, qualified electronic time stamps in § 41 and electronic registered delivery services in § 41 Regulation 910/2014 contains statements on the "presumption of the integrity" in Article 41 and also in Article 43 (see Sec. 4.4.3 for more details).

[117] In German "Beweiskraft" according to [70] or "Beweiswert" [152] which is often used synonymously in German.

[118] In German "gesetzliche Beweisregeln" [§ 286 (2) ZPO n.F. in German] or "statutory rules of evidence" [§ 286 (2) ZPO n.F. in the English translation].

[119] In German "gesetzliche Beweisvermutung" according to the PONS [398] dictionary `http://en.pons.eu/translate?q=Gesetzliche+Beweisvermutung&l=deen`.

[120] This thesis notes that this German legal text has just recently (18th of July 2017) been updated [99], but an earlier version [96] got analysed as well because older discussions in the legal debate are based on the former version.

[121] In German "Beweiswürdigung" according to [70].

[122] If the same literal text was also in ZPO a.F., as well as in ZPO n.F., this thesis will reference it as ZPO.

[123] The English translation was taken from the translation provided in [96]. Because this translation is based on an outdated version of the ZPO it was copied only after carefully checking that there is no literal difference in the German text when compared with the latest version [99].

The ZPO has two strains when evaluating the evidence, as laid out in § 286 ZPO:

> "(1) The court is to **decide, at its discretion and conviction**, and taking account of the entire content of the hearings and the results obtained by evidence being taken, if any, whether an allegation as to fact is to be deemed true or untrue. The judgment is to set out the reasons informing the conviction of the judges.
>
> (2) The court shall be **bound to statutory rules of evidence** only in the cases designated in the present Code."[124] [125] [§ 286 ZPO]

The above quote shows that the ZPO[126] offers two main legal concepts that govern the evaluation of evidence: *court's discretion and conviction* and *statutory rules of evidence*:

1. **Statutory rules of evidence** (discussed in Sec. 4.1.1):
   Following § 286 paragraph 2 ZPO "statutory rules"[125] [99] assign a probative value; in the case of electronic documents with a qualified electronic signature the statutory rules assign a prima facie evidence or evidence with a legal presumption of authenticity[127].
2. **Evaluation of evidence at the court's discretion and conviction:**
   This evaluation of evidence at the court's discretion and conviction can be done based on five different types of proofs: experts, visual inspections, interrogation of party[128], documents, or witnesses[129]. This thesis focus is electronic documents which cannot be interviewed, nor are the documents itself experts or witnesses. Hence, this thesis is interested in documents, which become *documentary evidence*. If, however, documentary evidence is not achieved then the probative value of electronic documents can still be permitted as *evidence taken by visual inspection*.
   - **Documentary evidence** or 'Urkundenbeweis'[130] following § 416 ZPO[131] is discussed in Sec. 4.1.2, and
   - **evidence taken by visual inspection** or 'Augenscheinsbeweis'[130] following § 371 ZPO[132] is discussed in Sec. 4.1.3.

The legal background on statutory evidentiary presumption will be provided next in Sec. 4.1.1; followed by two subsections 4.1.2 and 4.1.3 that discuss the concept of probative value for paper-based documents. Those concepts have been codified into the text of legal acts before electronic documents became more wide-spread. Hence, they have then been amended or extended to cover the electronic forms. In Sec. 4.2 the legal texts regarding the probative value for electronic documents are analysed; the conclusion of that analysis is that those legal texts set requirements that point to signature specific legal texts with more technical requirements.

4.1.1. Statutory evidentiary presumption following German ZPO

According to the German Code of Civil Procedure (ZPO) statutory evidentiary presumption means that the appraisal of evidence results in a probative value being adducted through an explicit legal rule. These "statutory rules of evidence" [ZPO] under certain defined circumstances[133] assign the signed document a high probative value.

[124] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[125] The English translation was taken from the translation provided in [96]. Because this translation is based on an outdated version of the ZPO it was copied only after carefully checking that there is no literal difference in the German text when compared with the latest version [99].

[126] Also § 286 ZPO, § 261 StPO, and § 108 VwGO.

[127] 'Anscheinsbeweis der Echtheit' oder 'gesetzliche Vermutung der Echtheit' translated to 'evidence with a legal presumption of authenticity' according to the PONS [398] dictionary `http://en.pons.eu/translate?q=Gesetzliche+Beweisvermutung&l=deen`.

[128] Not in German StPO.

[129] In German 'Sachverständige, Augenschein, Parteivernehmung, Urkunden, Zeugen' according to [70].

[130] German-English translation according to Langenscheidt Recht Englisch [70].

[131] Can also be found in for example § 249 StPO.

[132] Can also be found in for example § 86 StPO 'Richterlicher Augenschein '.

[133] For electronic documents they demand adherence to certain properties from signature legislation which are the subject of a thorough analysis presented in the following sections, e.g., in Sec. 4.2.2 of this chapter.

Statutory evidentiary presumption does not mean that the high probative value assigned cannot be reduced in a court case. In court, the probative value assigned can be questioned, e.g., after hearing expert witnesses. In the following this thesis differentiates the probative value assigned in accordance with the German ZPO by their different level of difficulty to rebut it:

- **Prima facie evidence**[134] is considered a high probative value. When a party, in order to contest the prima facie evidence, can lay out "facts giving rise to serious doubts" such that the probative value "can be cast into doubt" then it is no longer considered as evidence as stated in § 371a ZPO[135].
- **Evidence with a legal presumption of authenticity**[136] is considered the highest probative value. In accordance with § 437 (1) ZPO, to refute an evidence with a legal presumption of authenticity "its opposite" must be proven to the court as stated in § 292 ZPO[135].

4.1.2. Probative value of paper-based documents induced by documentary evidence ('Urkundenbeweis') following German § 416 ZPO

As a background, this section discusses the probative value for paper-based documents, before looking at electronic documents in Sec. 4.2. The probative value of documentary evidence, e.g., signed private records and documents, is described in the German Code of Civil Procedure (ZPO) as follows:

> "To the extent that private records and documents are signed by the parties issuing them, or have been signed using a mark that has been certified by a notary, they shall establish **full proof that the declarations they contain have been made by the parties who prepared such records and documents**."[137] [135] [§ 416 ZPO]

Hence, documentary evidence establishes full proof that the declarations, e.g., statements, contained within the signed private document are issued by the signatory. This is evidence with a legal presumption of authenticity.

4.1.3. Probative value of paper-based documents induced by evidence taken by visual inspection ('Augenscheinsbeweis') following German § 371 ZPO

Before looking at electronic documents in Sec. 4.2, this section discusses the probative value for paper-based documents. Evidence submitted to this type of probative value assignment process actually did not qualify for any other type because it missed some qualities required to be judged as another type of evidence.

> "Evidence taken by visual inspection
>
> (1) Evidence taken by visual inspection is offered by designating the object to be inspected visually and by citing the facts regarding which evidence is to be provided.
>
> [...]"[135] [§ 371 ZPO]

Additionally, the outcome of the analysis of the probative value of evidence taken by visual inspection depends on the source of the document. The ZPO differentiates between:

- documents of official authority ('Öffentliche Urkunden'[138]) and
- non-public documents executed by private party ('Privaturkunden'[138]).

[134] In German 'Anscheinsbeweis' (praesumptio iuris) according to Langenscheidt Recht Englisch [70].

[135] The English translation was taken from the translation provided in [96]. Because this translation is based on an outdated version of the ZPO it was copied only after carefully checking that there is no literal difference in the German text when compared with the latest version [99].

[136] In German 'gesetzliche Vermutung der Echtheit' according to Langenscheidt Recht Englisch [70].

[137] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[138] German-English translation according to Langenscheidt Recht Englisch [70].

The above separation is important, as it is a general separation done in the ZPO. Hence, the same differentiation is also important when judging the probative value of electronic documents. This thesis refrains from discussing and dissecting the probative value and statutory evidentiary presumption for paper documents here any further. Instead, in the following Sec. 4.2 this thesis returns to the focus of interest: Electronic documents.

4.2. Probative value and statutory evidentiary presumption of electronic documents in German and UK law

This section describes how the probative value is assigned to electronic documents and under which circumstances an electronic signature's probative value is assigned as statutory evidentiary presumption. First and foremost, the term 'electronic document' is defined. In this section the two legal notions of probative value and statutory evidentiary presumption for electronic documents are given on the basis of a concrete legal regulation: the German Code of Civil Procedure (ZPO) and the Fiscal Code of Germany (AO). Additionally, this section also takes a brief look at UK signature legislation.

4.2.1. Definition of electronic document

In German law the German term 'elektronische Signatur' can be found in several places, e.g., §§ 130a and 371a ZPO. It got introduced when the German Electronic Signature Act (SigG) was updated to implement Directive 1999/93/EC. In Germany, the "electronic document" covers more than only the digitised version of a paper document[139].

The thesis thus defines it technically as follows:

Definition 64 : Electronic Document

The electronic document refers to the message, as defined in Definition 2, that carries digitised information when it is referred to in a legal context.

Note 1 This is a legal term.

Note 2 The information contained in an electronic document is digitised and can be texts, pictures, sounds as well as videos following [46, p. 1016] and [219, p. 10].

4.2.2. German § 371a ZPO and § 87a AO on the probative value and statutory evidentiary presumption of electronic documents

§ 371a ZPO describes the "[e]vid्entiary value of electronic documents" [§ 371a ZPO][140] in civil proceedings. As for paper-based documents, the ZPO distinguishes between documents of official authority (§ 371a (1) ZPO) and non-public documents executed by private party (§ 371a (3) ZPO). Both are of interest, especially because one possible application of malleable signatures is to remove information from documents from a trusted source of origin.

[139] This thesis follows *Berger* who states"[...] ein weites Begriffsverständnis bezieht alle möglichen Inhalte ein, also neben Texten auch Grafik-, Audio- und Videodateien sowie Software. Der weiten Auffassung ist zu folgen." [46]. In addition *Grigorjew* states "Mittlerweile hat sich jedoch ein weites Verständnis, wonach in einem 'elektronischen Dokument' nicht Textdokumente, sondern auch sämtliche Medieninhalte wie zum Beispiel Grafik-, Audio- und Videodateien sowie Software enthalten sein können, durchgesetzt." [219, p. 10].

[140] The English translation was taken from the translation provided in [96]. Because this translation is based on an outdated version of the ZPO it was copied only after carefully checking that there is no literal difference in the German text when compared with the latest version [99].

EXAMPLE

Example: Consider a document of official authority which is in need of sanitization. Assume this governmental document contains state secrets, personal data or personally identifiable information[141]of citizens that must be de-classified before public release.[142] A public release of a redacted version might be needed if the public administration releases information under the German Federal Act Governing Access to Information held by the Federal Government (IFG)[143] or the US Freedom of Information Act (FOIA) [490]. An example can be seen in Fig. 27. Examples for non-public documents executed by private party are the supply chain messages (see Sec. 1.2.3).

An evaluation of evidence at the court's "discretion and conviction"[144] according to § 286 ZPO can take place for electronic versions for documents of official authority and non-public documents executed by private party. Hence, the individual judgement of assigning a higher probative value of electronic documents, i.e., if it is not assigned by statutory rules, by each court in each case remains possible. However, if statutory rules of evidence exist, then the assignment of the probative value by those rules, which leads to a statutory evidentiary presumption, has precedence over the outcome of the court's discretion [§ 286 (2) ZPO]. Analog to the already discussed Sec. 4.1.1 statutory rules for paper-based documents, the ZPO has also rules for the probative value of electronic documents in civil proceedings[145], e.g., § 371a ZPO. The statutory rules in the ZPO induce a probative value at the level of prima facie evidence or that of evidence with a legal presumption of authenticity.

The minimum assignment by the ZPO through statutory rules is prima facie evidence. To destroy the probative value of prima facie evidence, the court needs to be convinced "[...] only by facts giving rise to serious doubts [...]" [§ 371a ZPO] that the probative value "[...] can be cast into doubt [...]" [§ 371a ZPO]. This need for facts that induce serious doubts can also be found in § 87a paragraph 5 sentence 2 AO. In the following this thesis will use the AO to identify the change in language applied in German legal texts by the eIDAS-Durchführungsgesetz (eIDASDG) [98] to adopt to the fact that the VDG and Regulation 910/2014 have superseded the SigG:

> "The apparent authenticity, as verified pursuant to the Electronic Signature Act, of a document transmitted with a qualified electronic signature in accordance with the Electronic Signature Act **may only be called into question by way of facts which lead to serious doubt as to whether the document was transmitted in keeping with the will of the owner of the signature key**."[146] [§ 87a paragraph 5 sentence 2 of AO a.F.]

Following Article 6 eIDASDG paragraph 5 sentence 2 is adjusted to read as follows:

> "For the evidentiary value of electronic documents section 371a of the Code of Civil Procedure (ZPO) shall apply mutatis mutandis."[147] [§ 87a paragraph 5 sentence 2 of AO n.F.]

This can be interpreted as an idea that the German eIDASDG will not only remove the literal links to the SigG but also to centralise the descriptions of the rules concerning the probative value towards the ZPO as a single point. Still, as the rules stated in the AO a.F. were in accordance with the ZPO, the change following eIDASDG is not changing the analysed statements made by the ZPO regarding the probative value: See Sec. 4.2.2.1 and Sec. 4.2.2.2 for more details on the probative value induced by the the ZPO and how the eIDAS-Durchführungsgesetz (eIDASDG) changed the pointer in the ZPO from pointing to SigG to directly point to the Regulation 910/2014.

4.2.2.1. Probative value of electronic documents of official authority following German legislation (ZPO)

§ 371a paragraph 3 ZPO n.F. states that the same rules as for documents of official authority in paper apply when they are in electronic form:

[141] The term personally identifiable information is meant to be interpreted according to EU data protection legislation like [184].

[142] See also example in Sec. 1.2.3 and the statements made by the German Federal Administrative Court (BVerwG) in the ruling from 23.06.2011 - Az. 20 F 21/10, Randnummer 22.

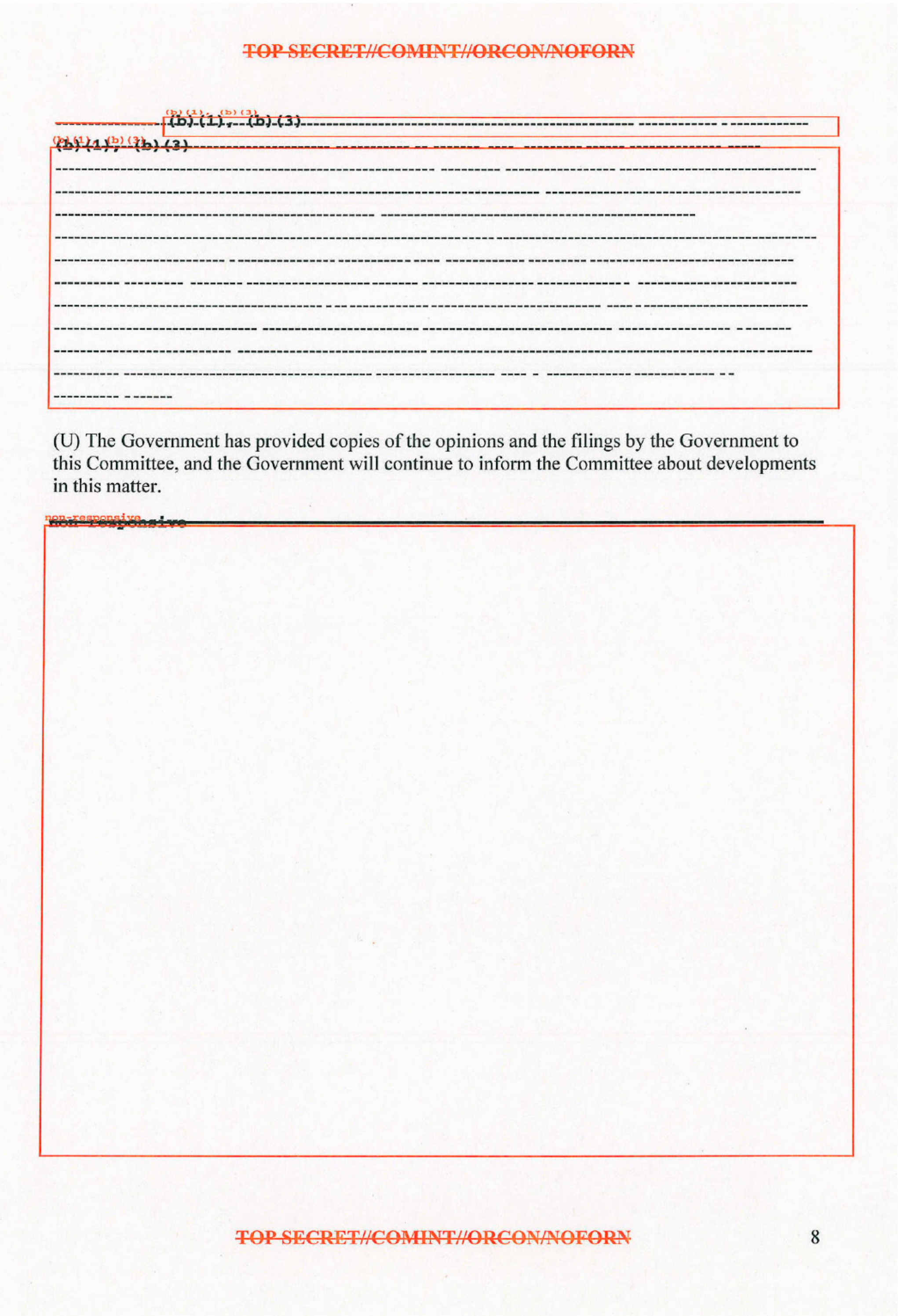
[143] In German 'Gesetz zur Regelung des Zugangs zu Informationen des Bundes (Informationsfreiheitsgesetz - IFG)'.

[144] The English translation was taken from the translation provided in [96]. Because this translation is based on an outdated version of the ZPO it was copied only after carefully checking that there is no literal difference in the German text when compared with the latest version [99].

[145] Translation from German 'Zivilprozess' to English according to Langenscheidt Recht Englisch [70].

[146] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[147] Translated from the German "Für die Beweiskraft elektronischer Dokumente gilt § 371a der Zivilprozessordnung entsprechend." [§ 87a paragraph 5 sentence 2 of AO n.F.].



~~TOP SECRET//COMINT//ORCON/NOFORN~~

(b)(1), (b)(3)

(b)(1), (b)(3)

(U) The Government has provided copies of the opinions and the filings by the Government to this Committee, and the Government will continue to inform the Committee about developments in this matter.

non-responsive

~~TOP SECRET//COMINT//ORCON/NOFORN~~ 8

Figure 27. Example for a redacted document of official authority from an answer to a US FOIA request; taken from [161]

“Evidentiary value of electronic documents

[...]

(3) The rules concerning the evidentiary value of public records and documents shall be applied mutatis mutandis to electronic documents created, in accordance with the requirements as to form (public electronic documents), by a public authority within the purview of its official authority, or by a person or entity of whom it is incontestably assumed that

> he or she is acting properly and in good faith within the sphere of business assigned to him or it. Where the document bears a **qualified electronic signature**, section 437 shall apply mutatis mutandis.
> [...]” [§ 371a ZPO n.F.][148] ,[149]

In § 371a paragraph 3 sentence 2 ZPO n.F. it is stated that in the case that the electronic document is signed with a qualified electronic signature, § 437 paragraph 1 ZPO n.F. should apply. The latter reads as follows:

> “Authenticity of German public records and documents
>
> (1) Records and documents which, by their form and content, appear to have been executed by a public authority or by a person or entity, shall be presumed to be authentic.
> [...]” [§ 437 paragraph 1 ZPO n.F.][150]

This thesis uses the term Signer for the entity that is “the holder of the signature key” [§ 371a (1) ZPO a.F.] or “person responsible” [§ 371a (1) ZPO n.F.] and is called “signatory” [ZPO]; refer to Sec. 2.10 for further definitions of involved entities and for the definitions of Signer (see Definition 18) as well as signatory (see Definition 19).

Documents of official authority with qualified electronic signatures have the highest probative value of evidence with a legal presumption of authenticity. Statutory rules laid out in the German ZPO award a document of official authority that carries a valid qualified electronic signature with an evidence with a legal presumption of authenticity. Following that the signed document establishes the full proof that the contents

- are issued by the Signer, and
- are correct, i.e., can be considered as facts[a], and
- are presumed to be authentic[b].

For a further in-depth discussion see *Roßnagel and Fischer-Dieskau* [415].

[a] §§ 415, 417 and 418 ZPO together with § 371a paragraph 3 sentence 2 ZPO.
[b] § 437 paragraph 1 ZPO together with § 371a paragraph 3 sentence 2 ZPO.

4.2.2.2. Probative value of electronic non-public documents executed by private party following German legislation (ZPO and AO)

For the probative value of documents of official authority it was codified in law early on and quite explicitly that public authorities shall be able to make and exchange legally binding statements by means of electronic documents[151]. This was not the case for electronic non-public documents executed by private party. § 371 ZPO got amended with § 371a to handle electronic non-public documents executed by private party[152]. In § 371a paragraph 1 sentence 1 ZPO it is stated that the same rules as for paper-based non-public documents executed by private party apply when they are in electronic form but **only** if they bear a valid “qualified electronic signature” [§ 371a paragraph 1 ZPO]. Before Regulation 910/2014 was in affect the ZPO pointed to the “German Electronic Signature Act (Signaturgesetz)” as follows:

> “Evidentiary value of electronic documents
>
> (1) The rules concerning the evidentiary value of private records and documents shall be applied mutatis mutandis to **private electronic documents bearing a qualified electronic signature**. The appearance of authenticity of a declaration available in electronic form, as obtained from reviewing it **pursuant to the Electronic Signature Act (Signaturgesetz)**, can be cast into doubt only by facts giving rise to serious doubts as to the declaration having been made by the holder of the signature key.

[148] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[149] Note, the translation is taken from the unchanged sentences of ZPO a.F. paragraph 2 [96]; the later versions (including the latest) of ZPO offered a revised paragraph 2 added to cater for changes introduced by the De-Mail legislation and also the original text from paragraph 2, now in paragraph 3, was appended; however the De-Mail changes do not change the probative value of electronically signed documents and thus is ignored in this analysis; see Sec. 4.2.2.3 for details.

[150] The English translation was taken from the translation provided in [96]. Because this translation is based on an outdated version of the ZPO it was copied only after carefully checking that there is no literal difference in the German text when compared with the latest version [99].

[151] *Rüßmann* is talking about this in reference to electronic property ownership records [424].

[152] More on that topic by *Viefhues* [494].

> [...]”[153] [§ 371a paragraph 1 ZPO a.F.]

With the SigG ceasing to have effect in July 2017 the Article 9 (14) of eIDAS-Durchführungsgesetz [98] changed the ZPO to point to Regulation 910/2014 instead:

> “Evidentiary value of electronic documents
>
> (1) The rules concerning the evidentiary value of private records and documents shall be applied mutatis mutandis to **private electronic documents bearing a qualified electronic signature**. The appearance of authenticity of a declaration available in electronic form, as obtained from reviewing it **pursuant to Article Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC**, can be cast into doubt only by facts giving rise to serious doubts as to the declaration having been made by the holder of the signature key.
>
> [...]”[154] [§ 371a paragraph 1 ZPO n.F.]

Also the Fiscal Code of Germany (AO) contained a direct pointer to signature legislation; namely in § 87a paragraph 4 AO a.F.. Before Regulation 910/2014 was in force the AO was pointing to SigG:

> “(4) Where it is stipulated by law that administrative acts or other measures of the revenue authorities be in written form, this may be replaced by electronic form unless otherwise required by law. In this event, the electronic document shall bear a qualified electronic signature in accordance with the Electronic Signature Act.” [§ 87a paragraph 4 sentence 1,2 AO a.F.]

In its latest version this pointer — as SigG has ceased to have effect — is removed. Following Article 6 eIDASDG paragraph 4 sentence 2[155] it currently reads as follows:

> “(4) Where it is stipulated by law that administrative acts or other measures of the revenue authorities be in written form, this may be replaced by electronic form unless otherwise required by law. In this event, the electronic document shall bear a qualified electronic signature.” [§ 87a paragraph 4 sentence 1,2 AO n.F.]

German legal texts when mentioning electronic documents point to EU signature legislation (eIDAS). The German Code of Civil Procedure (ZPO) explicitly mentions that you need a qualified electronic signature; and § 371a paragraph 1 sentence 2 ZPO directly points to the legislation regarding the mechanisms for electronic signatures. Namely, before Regulation 910/2014 it pointed to the German “Electronic Signature Act”. The referred to German “Electronic Signature Act” is the German Signaturgesetz SigG. SigG together with SigVO and the technical catalogue of recognised algorithms [84, 85] is the German legislation that implemented the former EU Directive 1999/93/EC, including technical details on the algorithms laid down in the catalogue of recognised algorithms [84, 85].

With Regulation 910/2014 superseding Directive 1999/93/EC this got updated and the German legal texts now directly point at the Regulation 910/2014, like in the ZPO n.F. quoted above. This was also adjusted in many other German legal texts that mentioned qualified electronic signatures by the eIDAS-Durchführungsgesetz [98]. For example the pointer to the German SigG was removed in the Fiscal Code of Germany (AO) [§ 87a paragraph 4 sentence 1,2 AO a.F.] as quoted above. Of course the overall requirement that “[...] the electronic document shall bear a qualified electronic signature.” [§ 87a paragraph 4 sentence 1,2 AO n.F.] remains.

Non-public documents executed by private party with qualified electronic signatures gain a high probative value as prima facie evidence. If a non-public document executed by private party carries a valid qualified electronic signature it can get awarded with an increased probative value under the regime of § 371 ZPO as evidence taken by visual inspection (“Augenscheinsbeweis”). As the statutory rules of evidence from the ZPO for paper documents apply, an electronic non-public document executed by private party with a qualified electronic signature is assigned prima facie evidence. This would allow to establish that the signed document’s contents

[153] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[154] The translation was based on the existing translation provided for ZPO a.F. [96]; **bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[155] In German “[...] 2. In Absatz 4 Satz 2 werden die Wörter ”nach dem Signaturgesetz” gestrichen.” [98].

- are issued by the Signer[a] , and
- are presumed to be authentic.

See *Roßnagel and Fischer-Dieskau* [415] for an in-depth discussion.

[a] This thesis uses the term Signer for the entity that is "the holder of the signature key" [§ 371a (1) ZPO a.F.] or the "person responsible" [§ 371a (1) ZPO n.F.] (see Sec. 2.10 for further definitions of involved entities).

4.2.2.3. The text added to § 371a ZPO due to De-Mail is not relevant for the research question

Note, in the latest version of the § 371a paragraph 2 ZPO was added to cater for changes introduced by the German De-Mail-G[156] legislation. When mentioning the use of digital signatures the German De-Mail legislation always uses the term "qualified electronic signature"[157]. Thus, this legislative text is also pointing to the notion of a qualified electronic signature specified by Regulation 910/2014 — or previously in SigG following Directive 1999/93/EC[158].

Paragraph 2 of § 371a ZPO n.F. under certain conditions[159] grants the electronic document sent in the De-Mail system a heightened legal probative value regarding its origin, i.e., the natural person that sent it[160]. It is out of scope to discuss the exact probative value of information exchanged through the German De-Mail system in this thesis; for further analysis suggested starting points are [153, 411, 417].

Note that paragraph 2 of § 371a ZPO n.F. does not change the probative value of electronically signed documents outside the De-Mail setting and thus is ignored in this analysis. While it is still out of scope to analyse if an RSS and SSS can be used in a De-Mail setting, it would be easier if it would in general supply a high probative value, i.e., also outside the De-Mail setting.

4.2.3. United Kingdom law on the probative value of electronic documents with a qualified signature

In the United Kingdom (UK) there is no direct legislation regarding the probative value for the electronic document as a whole. Alike all other European Member States, the UK introduced state legislation to implement the EU Electronic Signature Directive 1999/93/EC [175]: the Electronic Communications Act 2000 (ECA)[368]. Like the EU Directive the UK ECA is focussed solely on electronic signatures.

UK's ECA Section 7[368] is implementing Article 5.2 and is directed towards implementing Article 5.1.(b) of the EU Directive, according to a statement of the UK's Department for Business Enterprise & Regulatory Reform (BERR):

> "Article 5.2 is implemented into UK law through Section 7 of the Electronic Communications Act 2000." [476]

[156] See [97].

[157] In German it notes for example "[...] dauerhaft überprüfbaren qualifizierten elektronischen Signatur;" § 5 Absatz 5 De-Mail-G [97].

[158] Indeed the eIDASDG changed the De-Mail-G in that it removed the references to SigG; namely it instructs to drop the relevant clause by stating "[...] werden jeweils die Wörter "nach dem Signaturgesetz" gestrichen." [eIDASDG].

[159] Among the mentioned condition the following requirements are stated: (a) strong authentication of the natural person that is associated with the De-Mail account that the relevant person uses exclusively for its own (translated from the German "Hat sich eine natürliche Person bei einem ihr allein zugeordneten De-Mail-Konto sicher angemeldet "[§ 371a para. 2 ZPO n.F.]); (b) the De-Mail provider must be accredited, i.e., the provider must follow certain rules when registering and authenticating users and handling user's signature generation keys (translated following the German "Der akkreditierte Diensteanbieter [...]" § 5 Absatz 5 De-Mail-G [97]) and (c) the signature generated on behalf of the natural person must be a qualified electronic signature (translated following the German "Hierzu versieht er im Auftrag des Senders die Nachricht mit einer dauerhaft überprüfbaren qualifizierten elektronischen Signatur;[...] " § 5 Absatz 5 De-Mail-G [97]).

[160] *Roßnagel* [411] stated that it heightens the legal certainty; translated from German "De-Mail erhöht somit nicht nur die Rechtssicherheit[...] " [411]. However, *Roßnagel* [411] also highlights that a high probative value for the contents of a document are still only given by signing the contents using a qualified electronic signature; in German "Auf Grund der nachgewiesenen Vertrauenswürdigkeit der sicheren Anmeldung des identifizierten Nutzers mit zwei unabhängigen Sicherungsmitteln im Einzelfall dürften die Gerichte — ähnlich wie bei EC-Karten — einen widerlegbaren Anschein annehmen, dass der Nutzer selbst von seinem Konto aus im Internet gehandelt hat. Zu beachten ist allerdings, dass dieser Anscheinsbeweis nur die Identität des Handelnden betrifft, nicht jedoch die Inhalte von Willenserklärungen. **Sie sind beweissicher auch weiterhin nur mit qualifizierten Signaturen nachzuweisen**." [411].

> "Article 5.1(b) [...] requires Member States to ensure that qualified signatures are admissible in legal proceedings. This repeats the requirement of Article 5.2 for any electronic signature, and is implemented into UK law through Section 7 of the Electronic Communications Act 2000." [476]

UK's Electronic Communications Act 2000 (ECA) Section 7.-(1) reads as follows:

> "In any legal proceedings
> (a) **an electronic signature** incorporated into or logically associated related certificates with a particular electronic communication or particular electronic data, and
> (b) the certification by any person of such a signature, shall each be admissible in evidence in relation to any question as to the **authenticity** of the communication or data or as to the **integrity** of the communication or data." [368][161]

In the UK there are two main factors that determine what weight of evidence an electronic document will be given:

- the court's own assessment of the document, and
- any other statute or regulation that imposes a requirement of form on a document.

The latter requires the absence of an opposing statute or regulation that imposes a different requirement of form. To analyse this further would require an instantiated and detailed legal case to identify the applicable statues and hence is not done for this thesis. The other factor is evaluation of the document's evidence at the court's discretion and conviction, comparable to § 286 ZPO ("Freie Beweiswürdigung") [99]. This court's assessment of the document can take into account expert forensic evidence if the parties are contesting the integrity or authenticity of the document or its contents.

Regarding the question of statutory evidentiary presumption or the legal equivalence of electronic and handwritten signatures the UK's Department for Business Enterprise & Regulatory Reform (BERR) notes that:

> "There is no specific UK implementation of Article 5.1(a) as, under the law in England, Wales, Scotland and Northern Ireland, a hand written signature is already capable of being satisfied by an electronic one, including an advanced electronic signature." [476]

The UK saw no need for a dedicated regulation to fulfil the Electronic Signatures Directive 1999/93/EC and as such this has not been explicitly codified in UK law.

UK legislation allows a high probative value for signed electronic documents. To summarise, regarding the probative value Section 7 of UK ECA does confirm that "electronic signatures [...] shall [...] be admissible in evidence in relation to [...] the authenticity [...] or [...] to the integrity" [368]. Section 7 ECA does neither assign a certain probative value nor assess the relative weight of electronically signed electronic document in comparison to paper-based documents' evidence. Furthermore, Section 7 ECA makes no distinction about the type and strength of the electronic signature. In particular, as pointed out in [476] by UK's Department for Business Enterprise & Regulatory Reform (BERR), the statement of admissibility is not limited to advanced electronic signatures. Moreover, there is also no regulation that explicitly awards advanced electronic signatures that are based on a qualified certificate and are created by a secure-signature-creation device a higher probative value.

4.3. Analysis of legal definitions regarding integrity and authenticity from IT-related legislative acts in EU, German and US law

This section analyses text from of legislative acts that govern or mention the term Integrity and Authenticity with the goal to correctly define a mapping of the legal integrity notion into technical requirements in Chapter 7.

[161] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

It will discuss general EU information technology related legislative acts, the German Basic Right to “integrity of information technology systems” [100], and further existing legal definitions of the notion ‘Data Integrity’ given in EU regulation. Additionally, to widen the legal scope, Sec. 4.3.4 briefly analyses two acts from the United States of America (US) and their definitions of data integrity: HIPAA and FISMA.

4.3.1. EU law: Notion of integrity in EU legislative acts in the area of general IT

The European Union recognised that system complexity, technical failure, human mistake, accidents or attacks may all have negative consequences for the functioning and availability of the physical infrastructures that deliver important services to citizens [Directive 2009/140/EC]. In the EU, the national regulatory authorities should therefore ensure that the integrity and security of public communications networks are maintained[162]. Additionally, legislative acts on electronic signatures have been introduced, e.g., the former Directive 1999/93/EC of 1999 and Regulation 910/2014 in forth since July 2016. Both are discussed separately in Sec. 4.4.

Directive 2009/140/EC, which amends among others the Directive 2002/21/EC on electronic communication networks[163], introduces a distinction between network “integrity” and “security”[164]:

> “The national regulatory authorities shall promote the interests of the citizens of the European Union by inter alia: a) [...] f) ensuring that the **integrity** and security of public communications networks are maintained.” [Directive 2009/140/EC][165]

Article 13a of Directive 2009/140/EC[164] further differentiates this:

> “(1) Member States shall ensure that undertakings providing public communications networks or publicly available electronic communications services take appropriate technical and organisational measures to appropriately manage the risks posed to security of networks and services. Having regard to the state of the art, these measures shall ensure a level of security appropriate to the risk presented. In particular, measures shall be taken to prevent and minimise the impact of security incidents on users and interconnected networks.
> (2) Member States shall ensure that undertakings providing public communications networks take all appropriate steps to **guarantee the integrity of their networks, and thus ensure the continuity**.
> (3) Member States shall ensure that undertakings providing public communications networks or publicly available electronic communications services notify the competent national regulatory authority of **a breach of security or loss of integrity** that has had a significant impact on the operation of networks or services.”[165] [Directive 2009/140/EC]

After an interpretation Article 13a(2) EU Directive 2009/140/EC from Nov. 2009 maps to technical IT security terminology. EU legislators see “integrity” on one side as means to “ensure the continuity” [Article 13a(2) Directive 2009/140/EC] and on the other side they list “loss of integrity” and “breach of security” as distinct problems [Article 13a(3) Directive 2009/140/EC]. Both these usages of ‘integrity’ are not in line with the IT security definitions. However, this thesis suggests to interpret “security” in the above as meaning secrecy or ‘confidentiality’, which this thesis uses synonymously. ‘Confidentiality’ is a well known IT security term. In the same way, instead of “continuity” the technically better known term of ‘availability’ can be used. After the above adjustments Article 13a(3) would read: “a breach of ~~*security*~~ *confidentiality* or loss of integrity”[165]. With this interpretation in mind, Article 13a(2,3) could be interpreted to require that Member States shall ensure confidentiality, integrity and availability. These are the three classic IT security goals [249].

[162] See Recital 44 of Directive 2009/140/EC.

[163] Directive 2009/140/ECamends Directives 2002/21/EC on a common regulatory framework for electronic communications networks and services, 2002/19/EC on access to, and interconnection of, electronic communications networks and associated facilities, and 2002/20/EC on the authorization of electronic communications networks and services [180].

[164] Art. 4 lit. f and Chapter IIIa “Security and Integrity of Networks and Services”, Articles 13a, 13b, and Recital 28 of Directive 2009/140/EC.

[165] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

The implication between "integrity" and "availability", as given in Article 13a(2) of [Directive 2009/140/EC] "guarantee the integrity of their networks, and thus ensure the ~~*continuity*~~ *availability*"[a] [Article 13a(2) Directive 2009/140/EC], is technically correct if, and only if, loss or unavailability of data are also defined as integrity breaches. This is in line with computer security definitions, e.g., *Clark and Wilson* or RFC 4949. Hence, the provided rewording indicates that European legal texts are aligned with the technical definitions of IT security goals.

The final result of the analysis is given in Sec. 4.6.

[a] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

4.3.2. German law: Integrity as a constitutional basic right

The German Supreme Constitutional Court (BVerfG), the highest German court, in 2008 constituted a new basic right:

> "The general right of personality [...] encompasses the **fundamental right to the guarantee of the** confidentiality and **integrity of information technology systems**."[165] [100, Headnote 1]

This fundamental right is seen as a new basic or civil right. It protects the personal and private life of rights holders from the state accessing and modifying any IT devices. In particular, the German Supreme Constitutional Court stated in [100] that it wanted to protect citizens against unlawful covert state access to their IT systems. It targets the citizen's system as a whole, rather than offering only protection against state access to individual communications or certain stored data. The German Supreme Constitutional Court wanted to make sure that integrity of the IT system is not harmed by the state unless strong circumstances[166] are given and a court order[167] has been obtained.

The integrity notion of the German Supreme Constitutional Court given in 2008 does not tergiversate, but is different in scope and, thus, not directly applicable when assessing authorized modifications of electronic document. First and foremost, by constituting a basic right the highest German court acknowledged the importance of integrity protection of IT. The intention of this basic right is to allow a protection of data and IT systems from covert access [234] by the state. Here, the notion of integrity of the system is comparable to the technical definition from Biba (see Sec. 5.1.1). This thesis does not consider it useful to interpret the strong circumstances under which a violation of integrity is in accordance with the basic right — set forth by the German Supreme Constitutional Court (BVerfG) — as an authorization to a subsequent modification. Moreover, there are more reasons, why the German Supreme Constitutional Court's notion of integrity is not suitable to identify the influence of authorized changes on the probative value of signed data:

- The dealt-with attack is covert access to an IT system by a state attacker; the notion of integrity is geared to protect the system from working in an unwanted way to impede the secrecy of communication[a]. In a computer science notion this would mean that the system can be trusted to keep the confidentiality of messages.
- The protection scope is the IT system as a whole, not an electronic document of its own.
- Furthermore, the scope is the physical system at the end-point of a communication, e.g., personal computer, server, smartphones, etc., rather than the exchanged messages [234].

While to check a system's integrity would also require to some extent that messages exchanged with the system or messages stored within the system can be integrity-protected, this thesis will not further consider this new basic right when assessing authorized subsequent modifications of electronic document.

The final result of the analysis is given in Sec. 4.6.1.

[a] Compare the several references made that argue how the failure to protect integrity would result in an "encroachment" of the "secrecy of telecommunication" [100] that is guaranteed by German Basic Law.

[166] "if factual indications exist of a concrete danger to a predominantly important legal interest" [100, Headnote 2].

[167] "The secret infiltration of an information technology system is in principle to be placed under the reservation of a judicial order" [100, Headnote 3].

4.3.3. EU law: Notion of data integrity with respect to authorized modifications

This thesis found that different legal acts describe the notion of 'data integrity' differently. According to Regulation (EC) No 460/2004, by which the European Network and Information Security Agency (ENISA) was established, it is as follows:

Definition 65 : Integrity from Regulation (EC) No 460/2004

> *" "Data Integrity" means the **confirmation** that data which has been sent, received, or stored are complete and **unchanged**."[168] [179, Article 4 lit. f]*

This definition sees data integrity as the confirmation, e.g., output of some integrity checking mechanism.

Integrity is acknowledged as an essential part of ENISA's "network and information security" [179] in Article 4 lit. c of the Regulation (EC) No 460/2004 as follows:

> " "network and information security" means the ability of a network or an information system to **resist**, at a given level of confidence, accidental events or **unlawful or malicious actions that compromise the** availability, authenticity, **integrity** and confidentiality of stored or transmitted data and the related services offered by or accessible via these networks and systems."[168] [179, Article 4 lit. c]

Regulation (EC) No 460/2004 sees "integrity" as a state. According to Regulation (EC) No 460/2004 "integrity" [179] can be "compromised" [179, Article 4 lit. c] and it is a "confirmation"' [179, Article 4 lit. f], e.g., the outcome of a trusted process of integrity verification to gain confirmation of integrity.

Regulation (EC) No 460/2004 is ambiguous when mentioning modifications. Regarding authorized changes Regulation (EC) No 460/2004 is ambiguous, Article 4 literal f of the Regulation is explicitly not endorsing authorized modifications, while Article 4 literal c allows it. This is the result of a grammatical interpretation of the notion of authorized changes, ENISA's definition for data integrity requires data to be "unchanged" [179, Article 4 lit. f] to "resist [...] malicious actions" [179, Article 4 lit. c].

This contributes to the final Analysis Result 2 given in Sec. 4.6.2.

The Regulation 2016/679 [184] (GDPR) — more recent in its text than the 2004 regulation — explicitly mentions the term 'integrity' in its list of personal data protection principles in Article 5:

> "processed in a manner that ensures appropriate security of the personal data, including protection against **unauthorized** or unlawful **processing** and against accidental loss, destruction or damage, using appropriate technical or organisational measures ('**integrity** and confidentiality')."[168] [Article 5, para.1 (f) GDPR]

However, it makes no further reference to define the "technical [...] measures" [GDPR]. Neither does the GDPR point to any explicit technical standard.[169]

Further, the GDPR defines the term 'processing'[170] to include the "adaptation or alteration" [Article 4 (2) GDPR]. Hence, "unauthorized [...] processing" [Article 5, para.1 (f) GDPR] is interpreted here to not explicitly exclude authorized subsequent modifications.

[168] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[169] Compare also: "The Data Protection Directive does not make any specific reference to technical standards." [290].

[170] " 'processing' means any operation or set of operations which is performed on personal data or on sets of personal data, whether or not by automated means, such as collection, recording, organisation, structuring, storage, adaptation or alteration, retrieval, consultation, use, disclosure by transmission, dissemination or otherwise making available, alignment or combination, restriction, erasure or destruction;" [Article 4 (2) GDPR].

Regulation 2016/679 explicitly mentions "integrity" as a protection goal and lists "unauthorized [...] processing" as a threat. According to Regulation 2016/679 personal data shall not be "processed" [Article 5, para.1 (f) GDPR] such that " 'integrity and confidentiality' " are harmed. Note, they are quoted because this seems to be the short-hand for the personal data protection principle that is described. However, there are no pointers encouraging any explicit technical standards, which "[...] does not [...] imply that the legislator does not attribute any importance to standards in the context of the Directive." [290].

Implicitly, as the text does not explicitly exclude authorized processing, the GDPR allows authorized subsequent modifications.

4.3.4. US law: Notion of data integrity with respect to authorized modifications

In the following, two acts from US regulation are analysed. The US HIPAA is interesting because it explicitly calls the notion by its well-known name: "Integrity" [484]. The US FISMA is looked at because it is linked to a technical standards, the NIST's FIPS 199, which was written as a direct result of the legal act.

4.3.4.1. US HIPAA

The US Health Insurance Portability And Accountability (HIPAA) act was established in 1996. Following this, the US Department of Health & Human Services (HHS) codified some additional general and permanent rules and regulations that are often referred to as the *HIPAA Privacy Rule* and the *HIPAA Security Rule*. The latter, the Security Rule, was published in 2003 and is located in the Code of Federal Regulations (CFR)[171]. It defines Integrity in § 164.304 as follows:

Definition 66 : Integrity from US HIPAA Security Rule

> *"Integrity means the property that data or information have not been altered or destroyed in an unauthorized manner" [486, § 164.304 on p. 8376]*

It further explains in the implementation specification that to comply with HIPAA one needs integrity controls that "[i]mplement security measures to ensure that electronically transmitted electronic protected health information is **not improperly modified without detection** until disposed of."[172] [486, § 164.312 on p. 8379]

In § 164.312(c)(1) the HIPAA Security Rule [486] further states what the protection of integrity should avoid:

> "Integrity. Implement policies and procedures to protect electronic protected health information from **improper alteration** or destruction."[172] [486, § 164.312(c)(1)]

US HIPAA Technical Safeguard Standard implicitly tolerates authorized modifications. HIPAA explicitly mentions the term "improper" [486], which can be interpreted following the meaning of the word that HIPAA distinguishes between a modification that is not in accordance with accepted standards and a proper alteration. Hence, this thesis concludes that authorized, i.e., proper modifications would not constitute a breach of data integrity under HIPAA. Integrity is seen as being a property.

This analysis result contributes to Analysis Result 2.

4.3.4.2. US FISMA

The US Information Security Regulations Federal Information Security Management Act (FISMA) came into law in 2002 as Title III of the e-Government Act of 2002 [488]. The goal of the enactment is "[t]o enhance the management and promotion of electronic Government services and processes by establishing a Federal Chief Information Officer within the Office of Management and Budget, and by establishing a broad framework of measures that require using Internet-based information technology

[171] 45 CFR Part 160 and Subparts A and C of Part 164.

[172] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

to enhance citizen access to Government information and services [...]" [488]. NIST also stated that the FISMA "recognized the importance of information security to the economic and national security interests of the United States"[173] [356][354]. The text of FISMA prescribes that "adequate information security for all agency operations and assets" [488] should be provided.

Sec. 3542 with the title "Definitions" of US public law 107-347 states:

> "[...] (1) The term 'information security' means protecting information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide—
>
> > "(A) **integrity**, which means guarding against **improper information modification** or destruction, and includes ensuring information nonrepudiation and authenticity; [...]"[174] [488]

This shows that also in US regulation integrity is acknowledged as an information security goal. Within the US, the National Institute of Standards and Technology (NIST) used the FISMA's description and interpreted this in its standard FIPS[175] 199, as follows:

> "A loss of integrity is the **unauthorized modification** or destruction of information"[174] [354]

US FISMA implicitly tolerates authorized modifications. The definition of integrity given in FISMA and the following FIPS 199, it could implicitly allow *proper* or authorized modifications. However, it also lists two additional requirements that also authorized modifications would need to suffice: "ensuring information nonrepudiation and authenticity" [488]. Both, non-reputation and authenticity of origin relate to the goal of accountability, that is further analysed in following sections and in summary led to in Analysis Result 7.

This analysis result contributes to Analysis Result 2.

4.4. EU signature law: Notions of integrity and authenticity

In this section the following two European legal texts, which are specific to the domain of electronic signatures, are analysed:

Directive	1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures [Directive 1999/93/EC],
Regulation	(EU) No 910/2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC [Regulation 910/2014].

First, Directive 1999/93/EC was the current EU regulation when the work on the thesis started. It is the central EU legislative act on the probative value of digitally signed documents. Further, it is the source of the technical requirements as mentioned in member state regulations regarding the probative value in Section 4.1. Older court rulings and discussions are based on this regulation. Second, Regulation 910/2014 (Regulation 910/2014) was released in July 2014 as a reform of Directive 1999/93/EC. Regulation 2014/910, commonly also referred to as eIDAS, became active in September 2014 and became binding in member state legislation in June 2016. The idea is that Regulation 910/2014 (Regulation 910/2014) supersedes Directive 1999/93/EC:

> "The eSignature Directive (Directive 1999/93/EC) has been in place for over 12 years. The Directive has gaps, such as undefined obligations for national supervision of service providers, which are holding back cross-border eSignatures, and it does not cover many new technologies." [169]

It thus applies from 1 July 2016 in all member states. Still, this is very new and it will be seen if the forecasts on how eIDAS might influence the legal status of signatures in Germany made by *Jandt* [270] and also by *Roßnagel* [412] will be challenged and then decided in European courts.

[173] The American English spelling from original work is retained for terms and quotes.

[174] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[175] FIPS is short for 'Federal Information Processing Standards'.

For above reasons, this thesis will still analyse the previous legislative anchor for EU signature law, the Directive 1999/93/EC. However, as presented in the concluding Sec. 4.4.6 in which this thesis offers a grammatical analysis, there is no foreseen impact of the regulatory update by Regulation 910/2014 on the research questions, i.e., regarding modifications.

4.4.1. Terminology: Electronic vs. digital signature

Computer scientists as well as cryptographers might not be familiar with the term 'electronic signature' found in legal parlance, but with the term 'digital signatures'. In 1997, the German legislation, as well as the Italian and the US legislation, were established. The German statue enacted in 1997 was termed "Gesetz zur digitalen Signatur" and included in [88]. An overview of the history on signature legislation —prior to the EU Signature Regulations [175] of 1999 — is given by *Dumortier* [157]. One possible reason for the now widespread term of 'electronic' in many places of EU legislation is given by *Dumortier* [157]: The EU needed to introduce a EU wide legislation prohibiting each member state from regulating the legal status of signatures differently, and wanted to be 'technology-neutral'. As a consequence, legal terms like 'electronic document' and 'electronic signature' are found in many EU regulations instead of existing technically defined terms like 'digital signatures'. Member state legislation then adopted these terms.

For differentiation the term 'electronic signature' refers to the legal notion of the term 'digital signature' the cryptographic notion. Henceforth this thesis uses the term 'electronic signature' as defined in see Definition 30 whenever it refers to the legal notion. In contrast, this thesis uses the term 'digital signature' as defined in Sec. 2.9 to refer to the result of a cryptographic mechanisms called a digital signature scheme (DSS). More precisely the thesis understands a DSS as a 'digital signature scheme (DSS) based on asymmetric cryptographic techniques' to refer to a cryptographic scheme[a] that involves an asymmetric[b] cryptographic primitive[c]. The cryptographic value, i.e., the digital signature, produced by such a system is denoted by σ.

See also the analog use of 'electronic document' as a legal notion defined in Definition 64 and the term 'message' defined for cryptographic use in Definition 1.

[a] See Definition 7 for the term cryptographic scheme.
[b] See Definition 7 for the term asymmetric cryptographic techniques.
[c] See Definition 5 for the term primitive.

4.4.2. EU Directive 1999/93/EC: Goal and legal effect

EU Directives generally aim at the harmonisation of Member States' laws. The goal of Directive 1999/93/EC (Directive 1999/93/EC) can be best distilled from its recital 20:

> "Harmonised criteria relating to the legal effects of electronic signatures will preserve a coherent legal framework across the Community; national law lays down different requirements for the legal validity of handwritten signatures; whereas certificates can be used to confirm the identity of a person signing electronically; advanced electronic signatures based on qualified certificates aim at a higher level of security; **advanced electronic signatures which are based on a qualified certificate and which are created by a secure-signature-creation device can be regarded as legally equivalent to handwritten signatures only if the requirements for handwritten signatures are fulfilled**;"[176] [Directive 1999/93/EC]

Further, Article 5 paragraph 1 Directive 1999/93/EC describes the legal effects it shall has for electronic signatures:

[176] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

> "1. Member States shall ensure that advanced electronic signatures which are based on a qualified certificate and which are created by a secure-signature-creation device:
> (a) **satisfy the legal requirements of a signature in relation to data in electronic form in the same manner as a handwritten signature satisfies those requirements in relation to paper-based data**; and
> (b) **are admissible** as evidence in legal proceedings."[177] [Art. 5 para. 1 Directive 1999/93/EC]

Directive 1999/93/EC prescribes that member state legislation shall treat electronic signatures legally equivalent to handwritten signatures when judging the document's probative value, but only if the document's signature was created according to the highest standards. Directive 1999/93/EC results in Member States to adapt or amend their legal texts as needed to to allow electronic documents bearing electronic signatures to be admissible in court. However, Directive 1999/93/EC does not specify the achieved probative value [280, 284][a]. Directive 1999/93/EC does not give explicit rules of evidence for signed electronic signatures. Namely, Art. 5 para. 1 lit. a Directive 1999/93/EC states the achievable legal equality between a handwritten signature and an advanced electronic signature. Henceforth, this will be referred to as Directive 1999/93/EC's equality statement.

Following the equality statement of Directive 1999/93/EC and based on the German legal texts (as discussed in Sec. 4.2.2) the following probative value can be assigned: A valid handwritten signature of a natural person under a document allows to treat the document, if its content and form suffices, as a non-public document executed by private party. Then a statutory evidentiary presumption of a high probative value, namely prima facie evidence [284, 415], can be awarded. Hence, an advanced electronic signature on an electronic document that has suitable form and content gives statutory evidentiary presumption and a high probative value. Following the German legal texts the assigned probative value for electronic documents can range from prima facie evidence — for aforementioned non-public documents executed by private party — to evidence with a legal presumption of authenticity. The latter is awarded to a documents of official authority with an advanced electronic signature.

Finally, note that the the Directive 1999/93/EC's equality statement explicitly requires an electronic signature with specific security requirements in Art. 5 para. 1 Directive 1999/93/EC. It thus harmonises that statutory evidentiary presumption can only be awarded if the electronic signatures is created with a secret signature key for which the Signer has a qualified certificate and if it has been created by a secure-signature-creation device. See also *Balfanz and Laue* [22] and *Roßnagel and Fischer-Dieskau* [415] for a further legal analysis.

[a] *Klein* states in German "Allerdings stellt Art. 5 der Signaturrichtlinie [Directive 1999/93/EC] keine konkreten Anforderungen an das Maß der Beweiskraft, die elektronischen Dokumenten zukommen soll. Es wird hier nicht die Klassifikation der elektronischen Signatur als ein bestimmtes Beweismittel geregelt, sondern nur die generelle Verpflichtung der Zulassung als Beweismittel." [284] and *Klein* in [284] cites *Kilian* [280] as source for this argument.

4.4.3. EU Regulation 910/2014: Goal and legal effects

In Section 4 Regulation 910/2014 [182] the legal effects that electronic signatures shall have are captured as follows:

> "Article 25
> Legal effects of electronic signatures
>
> 1. An electronic signature shall not be denied legal effect and **admissibility** as evidence in legal proceedings solely on the grounds that it is in an electronic form or that it does not meet the requirements for qualified electronic signatures.
>
> 2. **qualified electronic signature shall have the equivalent legal effect of a handwritten signature**.
>
> [...]"[177] [Art. 25 Regulation 910/2014]

Article 3 Regulation 910/2014 gives definition of terms: signatory and electronic, advanced electronic and qualified electronic signatures. The term signatory in Definition 19 as well as the term qualified electronic signatures in Definition 32 were already defined in alignment with Article 3 Regulation 910/2014.

[177] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

From the incremental definition of the different levels of signatures in Article 3 Regulation 910/2014 this thesis deducts that the “qualified electronic signature” [Regulation 910/2014] combines the requirements on the “electronic signature” [Regulation 910/2014], on the “advanced electronic signature” [Regulation 910/2014][178], as well as on a qualified certificate[179].

A “qualified electronic signature” [Regulation 910/2014] must fulfil all the requirements set forth in Regulation 910/2014 and related legal texts. Regulation 910/2014, like Directive 1999/93/EC, allows electronic signatures who meet the high requirements of a “qualified electronic signature” [Art. 25, para. 2 Regulation 910/2014] to have the same effect as handwritten signatures. Article 25 para. 1 ensures, and this has also been in Directive 1999/93/EC[180], that a court can be asked to inspect and then judge the admissibility as evidence of an electronic signature even when “[...] it does not meet the requirements for qualified electronic signatures” [Regulation 910/2014].

Regarding the legal effect[181] and the probative value assigned, Regulation 910/2014 induces the same legal effect as Directive 1999/93/EC. A “qualified electronic signature” is equal to that of a handwritten signature. In his analysis *Roßnagel* [412, p. 3691] stated that Article 25 does not contain a discussion of the probative value. On first sight contrary, *Jandt, Michalek, and Dietrich* note that Regulation 910/2014 introduced explicit rules of evidence[182]. [271]. However, Regulation 910/2014 only for qualified electronic seals in § 35, qualified electronic time stamps in § 41 and electronic registered delivery services in § 41 contains statements on the “presumption of the integrity” [Regulation 910/2014]. In the same line of argumentation *Roßnagel* [412] noted an important change in the process of assignment of the probative value for seals following German legislation. Seals are intended as signatures for legal persons[183]. A seal on a document would elevate its probative value to that of legal presumption[184] [412, p. 3691]. Thus, qualified electronic seals would give evidence with a legal presumption of authenticity or legal presumption[185] the same high probative value as documents of official authority signed by an advanced electronic signature. Evidence that has been classified as evidence with a legal presumption of authenticity needs to be proven wrong in court. Following *Roßnagel and Fischer-Dieskau* [415] this is hard to do for qualified electronic signatures. Hence, Regulation 910/2014 gives a higher probative value — if not the highest: evidence with a legal presumption of authenticity — to electronic documents with a qualified electronic seal than to electronic documents with a qualified electronic signature — prima facie evidence. *Jandt, Michalek, and Dietrich* note that it still remains to see how these rules of evidence will be handled by courts [271, p. 689].

Regulation 910/2014 gives statutory evidentiary presumption if the document’s signature was created according to the highest standards. In Article 25 paragraph 2 Regulation 910/2014 [182] declares statutory evidentiary presumption and assigns a high probative value. This is done by stating the achievable legal equality between a handwritten signature and a so-called “qualified electronic signature” [Art. 25 para. 2 Regulation 910/2014]. Statutory evidentiary presumption is only awarded if the electronic signatures is created with a secret key for which the Signer has a “qualified certificate” [182] (following Article 28 of Regulation 910/2014) and if the electronic signature has been created by a “qualified electronic signature creation devices” [182] (following Article 29 and Annex II of Regulation 910/2014 [182]).

[178] More details on the requirements for “advanced electronic signatures” [Regulation 910/2014] are given in Article 26-29 and in Annex II Regulation 910/2014.

[179] See Sec. 2.11 for the definition.

[180] In Art. 5, para. 2 Directive 1999/93/EC.

[181] ’Rechtswirkung’ in German, translation according to Langenscheidt Recht Englisch [70].

[182] In German “Ein Novum der eIDAS-VO ist, das in europäischen Vorschriften explizite Beweisvorschriften aufgenommen worden sind.” [271].

[183] In Regulation 910/2014 ’signatures’ are created by a natural person and ’seals’ are created by a legal person.

[184] ”Diese Vermutungen gehen hinsichtlich des Objekts und der Wirkung weit über den Anscheinsbeweis für qualifizierte Signaturen nach § 371a ZPO hinaus. Sie gelten nicht nur für das Beweismittel, sondern auch für Tatsachen. Sie sind nach § 292 ZPO nur durch den Beweis des Gegenteils widerlegbar, nicht bereits durch Erschütterung der Anscheinsgrundlage.” [412, p. 3691].

[185] In German also translated with “gesetzliche Echheitsvermutung” [284].

4.4.4. EU Directive 1999/93/EC: Notions of integrity and accountability

The legislative body produced descriptions of processes and organisational requirements which an electronic signature system has to meet to become trusted to the above extent of statutory evidentiary presumption. This section looks at the requirements for the signature creation devices given in Annex III and the secure signature verification process as laid out in Annex IV of Directive 1999/93/EC.

Article 2 of Directive 1999/93/EC offers the definition of the term "advanced electronic signature" [175], stating the following requirements:

> "1. "electronic signature" means data in electronic form which are attached to or logically associated with other electronic data and which serve as a method of authentication;
> 2. "advanced electronic signature" means an electronic signature which meets the following requirements:
> (a) it is **uniquely linked to the signatory**;
> (b) it is capable of **identifying the signatory**;
> (c) it is created using means that the signatory can **maintain under his sole control**; and
> (d) **it is linked to the data to which it relates in such a manner that any subsequent change of the data is detectable**;" [175]

Directive 1999/93/EC demands the detectability of any subsequent change. Note, that Directive 1999/93/EC lacks a technically clear definition of the term Data Integrity and never defines it explicitly. However, instead, Article 2 (d) Directive 1999/93/EC explicitly states that the advanced signature must allow detecting any change the signed data was subjected to after it was signed.

Further, Article 2 Directive 1999/93/EC also makes sure that a compliant signature scheme enables to link signatures with the "signatory" [175]. For EU legal acts the "signatory means a person who holds a signature-creation device and acts either on his own behalf or on behalf of the natural or legal person or entity he represents" [175]. Hence, a lot of further clauses deal with certification and linkage of persons or entities with their signature-creation-data or their signature-verification-data. If the electronic signature is implemented by a classic digital signature scheme this can technically be done by public key certificates and a PKI. This PKI then establishes the link between the natural person and the cryptographic public key that is in the certificate, as requested per Article 2 paragraph 2.(a). Hence, this thesis has defined public key certificates in Sec. 2.11 based on Directive 1999/93/EC.

Article 2 (c) further requires the the Signer to be given the possibility to keep the signing key under his control, i.e., it must not be usable by any other party [410]. Especially, Article 2 (b) Directive 1999/93/ECrequires a link to the signatory, which is technically and organisationally established by means of public key infrastructures and legally recognised issuers of qualified electronic certificates (as defined in Sec. 2.11). Note, the question on how — or if at all — sole control can be exercised is a legal debate of its own[186], and out of scope for this thesis.

Directive 1999/93/EC requires a link between the Signer as a natural person (or their pseudonym) and the cryptographic material of the Signer to award statutory evidentiary presumption. The establishment of this link is assumed to exist with sufficient legal strength, but out of scope. § 2 2(a), 2(b) and 2(d) Directive 1999/93/EC require that the advanced electronic signature must allow linking the signed data with the signatory; a natural person (not a legal entity [§ 126a BGB Rd. 15, 435]) acting either on his own behalf or on behalf of the natural or legal person or entity he or she represents [§ 2 para. 3 Directive 1999/93/EC]. If a pseudonym is used, the issuer of the qualified certificate, shall reveal the true identity of the person only to the prosecution[a] [§ 126a BGB Rd. 14, 435]. For this thesis the entity holding the secret signature creation keys is denoted as the Signer. The discussion if the link between the bitstring, which represents the secret key technically, and the natural person that is associated with it — directly or in its role of the representative of a legal person — is

[186] *Mason* [330] offers a cautious position and warns about potential risks and incapabilities. However, *Mason* states "Whilst it is abundantly clear that a manuscript signature can be forged, or [...] that documents can be altered after they had been signed, digital signatures can help to resist attacks of these kinds." [330]. This leads to a reminder to be aware of potential risks as "[p]ractical problems [...] continue to exist with the implementation of a digital signature. However, the essential functions [...] can, largely, be met by the application of cryptography to the formation of a digital signature. As with manuscript signatures, there are always risks attached to the use of any form of electronic signature, and the user, whether a sending party or a receiving party, should make themselves aware of the risks before using any form of electronic signature for high value transactions." [330].

technically established is out of scope. The thesis assumes that current legally accepted technical and organisational means are to be used for all electronic signature, MSS as well as classical ones. This is stated as Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) in Sec. 3.8.3.

[a] German-English translation according to Langenscheidt Recht Englisch [70].

4.4.5. EU Directive 1999/93/EC: Technical process for signature generation

Annex III of Directive 1999/93/EC [175] describes the technical process and the systems to create signatures, as mentioned in Article 2 (d) Directive 1999/93/EC, by defining a secure-signature-creation device (SSCD). Also, Annex III mentions that the secrets needed to produce signatures, e.g., the $\mathsf{pk}_{\mathsf{sig}}$, are indeed secret to begin with [410] and can be kept secret during the creation of the signature, such that the resulting signature cannot be forged without access to the secrets. Additionally, the data to be signed by the device should not be manipulated by the device itself during the signing process. Annex III lists the following:

> "ANNEX III
> Requirements for secure signature-creation devices
>
> 1. Secure signature-creation devices must, by appropriate technical and procedural means, ensure at the least that:
> (a) the signature-creation-data used for signature generation can practically occur only once, and that their secrecy is reasonably assured;
> (b) the signature-creation-data used for signature generation cannot, with reasonable assurance, be derived and **the signature is protected against forgery using currently available technology**;
> (c) the signature-creation-data used for signature generation can be reliably **protected by the legitimate signatory against the use of others**.
>
> 2. Secure signature-creation devices **must not alter the data to be signed** or prevent such data from being presented to the signatory **prior to the signature process**."[187] [Directive 1999/93/EC]

A technical example for "signature-creation-data" [175]: the private key of an asymmetric key pair used in a digital signature scheme would be signature-creation-data as it is required to generate valid signatures. This key must remain usable only by the Signer who is the only party allowed to produce signatures. It must be kept secret and must not be usable by any other party (see also *Roßnagel* [410]).

Directive 1999/93/EC further gives recommendations for secure signature verification:

> "ANNEX IV
> Recommendations for secure signature verification
>
> During the signature-verification process it should be ensured with reasonable certainty that:
> (a) **the data used for verifying the signature correspond to the data displayed to the verifier**;
> (b) the signature is reliably verified and the result of that verification is correctly displayed;
> (c) the verifier can, as necessary, reliably establish the contents of the signed data;
> (d) the authenticity and validity of the certificate required at the time of signature verification are reliably verified;
> (e) the result of verification and the **signatory's identity are correctly displayed**;
> (f) the use of a **pseudonym is clearly indicated**; and
> (g) **any security-relevant changes can be detected**."[187] [Directive 1999/93/EC]

[187] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

Only at first sight, Annex IV (g) of Directive 1999/93/EC seems to ease the previous strict requirement to detect any subsequent change. However, because Annex IV (b) already mentioned the signature's verification outcome, this thesis sees this to be directed at the whole signature-verification process. Hence, the requirement of Annex IV (g), that "any security-relevant changes can be detected" [175] is rather meant as a recommendation towards the security of the systems involved during the process of signature verification.

Directive 1999/93/EC requires a secure signature-creation device (SSCD) to protect the Signer's private key during signature generation to award statutory evidentiary presumption. Annex III of Directive 1999/93/EC and Article 2 (c) Directive 1999/93/EC both require that the SSCD protects the information necessary for generating valid signatures, e.g., secret signature generation key $\mathsf{pk}_{\mathsf{sig}}$, from getting known to or used by parties other than the Signer.

4.4.6. EU Regulation 910/2014: Notions of integrity and accountability

The goal of Regulation 910/2014 [182] is to update and adjust Directive 1999/93/EC.[188] Next, this thesis will quickly highlight the differences in texts found in Regulation 910/2014 that describe the concept of Integrity and Accountability. Moreover, the impact of these changes within the scope of allowing authorized modifications on the probative value is analysed in this chapter.

Article 26 Regulation 910/2014 contains the following updated text (underlined) when compared to Article 2 Directive 1999/93/EC:

> "Article 26
> Requirements for advanced electronic signatures
> An advanced electronic signature shall meet the following requirements:
> (a) it is **uniquely linked to the signatory**;
> (b) it is **capable of identifying the signatory**;
> (c) it is created using electronic signature creation data that the signatory can, with a high level of confidence, ~~maintain~~ use under his sole control; and
> (d) it is **linked to the data signed therewith in such a way that any subsequent change in the data is detectable**."[189] [Regulation 910/2014 compared to Directive 1999/93/EC]

The requirements (a), (b) and (d) are worded exactly as in Directive 1999/93/EC. Requirement (c) is re-worded and has been amended by "with a high level of confidence" and instead of the Directive 1999/93/EC "maintain" Regulation 910/2014 has the notion of "use". This thesis interprets this as Regulation 910/2014 offering more flexibility. This is in line with *Roßnagel* [412] and *Kment* [286]. Namely, this flexibility allows remote signature creation services, or the "usage of the cloud saved private key"[286]. The security implications of the extended trust into the remote signature creation needs to be further discussed, it can be viewed as a reduction of technical security requirements as noted by *Jandt, Michalek, and Dietrich* [271][190]. The change of wording between the Directive and the Regulation is discussed in further detail in Sec. 4.4.7 and Sec. 4.4.8. However, the requirements regarding linking, identification and especially detection remain unchanged.

Regulation 910/2014 still requires detecting "any subsequent change" and linking to the Signer. The rewording and loosening of some conditions of Regulation 910/2014 does not impact on the requirements of detection of subsequent modifications and authentication of the signatory. Regulation 910/2014 Article 26 (d) Regulation 910/2014 clearly states that the advanced electronic signature al-

[188] "Why not just update the eSignature Directive? The eSignature Directive (Directive 1999/93/EC) has been in place for over 12 years. The Directive has gaps, such as undefined obligations for national supervision of service providers, which are holding back cross-border eSignatures, and it does not cover many new technologies. " [169, p. 3]

[189] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[190] *Jandt et al.* state in German "Im Vergleich zur fortgeschrittenen Signatur in Signaturrichtlinie (SigRL) und Signaturgesetz stellt die eIDAS-VO allerdings geringere Sicherheitsanforderungen." [271, p. 688] and cite previous own work [270]. Further they argue that the permission of remote signature creation reduces the security factor of physical possession, i.e. the physical power of disposition (in German "physikalische Verfügungsgewalt" [271, p. 689]).

lows to detect any change the signed data was subjected to after it was signed. Article 26 (a) and (b) Regulation 910/2014 are equal to Article 2 Directive 1999/93/EC and no contradicting statements are made, consequently the analysis results obtained previously for Directive 1999/93/EC in Sec. 4.4.4 equally apply to Regulation 910/2014.

4.4.7. EU Regulation 910/2014: Technical process for signature creation and comparison to Directive 1999/93/EC

Regarding the hardware and software that create signatures this thesis found that Regulation 910/2014 in Article 29 Regulation 910/2014 has the same technical requirements as Directive 1999/93/EC. Regulation 910/2014 now calls the hardware "qualified electronic signature creation devices" [Regulation 910/2014] (QSCD) whereas Directive 1999/93/EC calls it secure signature-creation device (SSCD):

> "Article 29
> **Requirements for qualified electronic signature creation devices**
>
> 1. Qualified electronic signature creation devices shall meet the requirements laid down in Annex II.
> 2. The Commission may, by means of implementing acts, establish reference numbers of standards for qualified electronic signature creation devices. Compliance with the requirements laid down in Annex II shall be presumed where a qualified electronic signature creation device meets those standards. Those implementing acts shall be adopted in accordance with the examination procedure referred to in Article 48 (2)." [Article 29 Regulation 910/2014]

The requirements for the process and the software and the hardware devices involved is either those of Annex II or codified via technical standards by additional acts. From the definitions in Article 2 it is observable that Regulation 910/2014 talks about a "qualified electronic signature creation device" [Article 2 (23) Regulation 910/2014] (QSCD) which is defined as follows:

> "(23) 'qualified electronic signature creation device' means an electronic signature creation device that meets the requirements laid down in Annex II;" [Regulation 910/2014]

The mentioned Annex II Regulation 910/2014 then describes the requirements and the first two sections are worded closely to what Annex III of Directive 1999/93/EC stated on the same matter. However, as Regulation 910/2014 is also taking into account that third parties might issue signatures on behalf of the signatory, called "qualified trust service providers", it got amended.

Next, this thesis gives the grammatical comparison between Annex II Regulation 910/2014 and Annex III of the former Directive 1999/93/EC to find if any changes introduced in the new EU regulation impact on the requirement of secure signature-creation device (SSCD) or alike, i.e., a secure smart card. In the following the additions of Regulation 910/2014 are highlighted by underlining and wording that was removed, but was present in Directive 1999/93/EC, is marked by strikeout. To not impact readability the removal of the hyphen in the writing of "signature-creation device" to "signature creation device" is not recorded below.

> "ANNEX ~~II~~III
>
> Requirements for ~~secure~~ qualified electronic signature creation devices
>
> 1. ~~Secure~~ Qualified electronic signature creation devices ~~must~~ shall ensure, by appropriate technical and procedural means, ~~ensure~~ that at least ~~that~~:
>   (a) the confidentiality of the electronic signature creation data used for electronic signature creation is reasonably assured;
>   ~~(a)~~ (b) the electronic signature creation data used for electronic signature ~~generation~~ creation can practically occur only once ~~, and that their secrecy is reasonably assured~~;
>   ~~(b)~~ (c) the electronic signature creation data used for electronic signature ~~generation~~ creation cannot, with reasonable assurance, be derived and the electronic signature is reliably protected against forgery using currently available technology;
>   ~~(c)~~ (d) the electronic signature creation data used for electronic signature ~~generation~~ creation can be reliably protected by the legitimate signatory against ~~the~~ use by others.

2. ~~Secure~~ Qualified electronic signature creation devices ~~must~~ shall not alter the data to be signed or prevent such data from being presented to the signatory prior to signing.
3. Generating or managing electronic signature creation data on behalf of the signatory may only be done by a qualified trust service provider.
4. Without prejudice to point (d) of point 1, qualified trust service providers managing electronic signature creation data on behalf of the signatory may duplicate the electronic signature creation data only for back-up purposes provided the following requirements are met:
   (a) the security of the duplicated datasets must be at the same level as for the original datasets;
   (b) the number of duplicated datasets shall not exceed the minimum needed to ensure continuity of the service."[191] [Regulation 910/2014]

In general, the term "electronic" is added to Regulation 910/2014. This makes Regulation 910/2014 more precise than Directive 1999/93/EC's text, by making explicit the applicability to electronic signatures. Further, all mentions of the term 'SSCD' have been adapted to 'QSCD'. Furthermore, the process of signing is called "creation" in Regulation 910/2014 and no longer "generation" [Directive 1999/93/EC], which this thesis interprets as no change in meaning.

Annex II section 1 — as well as section 2 — Regulation 910/2014 contain a "shall" [Regulation 910/2014] instead of the previous "must" [Directive 1999/93/EC]. The English writing standards used in Community legislation given by the European Commission's Directorate-General for Translation indicate that 'shall' is a positive command for imperative terms [173, 8.17-8.25]. Hence, Regulation 910/2014 'shall' containing statements are still interpreted by this thesis as statements of a requirement. Thus, again there is no change compared to Directive 1999/93/EC.

Further, Regulation 910/2014 uses the technically more often used term of confidentiality in its own subparagraph 1.(a) rather than "secrecy" [Directive 1999/93/EC] that was the second part of 1.(a) in the Annex of Directive 1999/93/EC. This thesis sees the notions of secrecy and confidentiality as synonymously. Thus, again there is no change compared to Directive 1999/93/EC.

Moreover, section 1 Regulation 910/2014 contains the contents from Directive 1999/93/EC, but they have been split over four sub subparagraph instead of three. In effect Regulation 910/2014 has split Directive 1999/93/EC's first bullet into two. Subparagraph 1.(b) Regulation 910/2014 contains same text the beginning of 1.(a) from Directive 1999/93/EC's Annex. Subparagraph 1.(d) Regulation 910/2014 contain the updated but otherwise same contents as in Directive 1999/93/EC's Annex. The same for subparagraph 1.(c) Regulation 910/2014 and except for an added "reliably" [Regulation 910/2014]. This added "reliably" to the "protected" strengthens subparagraph 1.(c) Regulation 910/2014 and it now reads alike 1.(d) Regulation 910/2014. It strengthens it, because legally "protected" would have allowed in general any technical protection, even if it can be circumvented[192], but "reliably protected" means that the technical protection is required to offer an adequate level of protection against attackers[193].

Finally, subparagraph 3 and 4 of Annex II Regulation 910/2014 are new compared to Directive 1999/93/EC. Both deal solely with the use of QSCD by trust services and do not influence this thesis's analysis regarding execution of malleable signature algorithms in hardware security modules, like secure smart cards.

Regulation 910/2014 has the same technical requirements on the hard-/software (QSCD) that generates the signature as Directive 1999/93/EC, but operation can be delegated. Regulation 910/2014 updated the member states' regulation in July 2016, however detailed impacts will only be seen in the future. The analysis given above in Sec. 4.4.7 shows that the changes and additions, with respect to the thesis's research question, are minor. The noteworthy permission of delegation of oper-

[191] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[192] As an example for a protection that is legally accepted but easy to technically circumvent is the DVD copy protection, e.g., "[i]n this new regime [referring to the DCMA], copy protection systems do not even have to be very good" [135].

[193] See *Orthacker, Centner, and Kittl* for a discussion on the "reliable protection of the signature-creation data" [365] in the light of Directive 1999/93/EC.

ation of the QSCD does not affect the requirement to execute each individual signature on an QSCD with equal requirements to that of an secure signature-creation device (SSCD). Hence, the "qualified electronic signature creation device" [Article 2 (23) Directive 1999/93/EC] (QSCD), comes with the same requirements as the SSCD in Directive 1999/93/EC.

The permission of remote signatures does not rule out RSS and SSS schemes, hence is not seen as an negative impact with respect to the research question of this thesis.

4.4.8. EU Regulation 910/2014: Remote signatures in the light of subsequent authorized modifications

One change introduced to EU signature legislation by Regulation 910/2014 is allowing "to entrust" [Regulation 910/2014] the qualified signature-creation device (QSCD) to a third party, which operates it on behalf of the Signer.

> Recital "(51)
> It should be possible for the signatory to entrust qualified electronic signature creation devices to the care of a third party, provided that appropriate mechanisms and procedures are implemented to ensure that the signatory has sole control over the use of his electronic signature creation data, and the qualified electronic signature requirements are met by the use of the device." [Regulation 910/2014]

Continuing on the matter, Recital 52 calls the signatures created as described in Recital 51 "remote signatures" [Regulation 910/2014].

This ability has been introduced to Regulation 910/2014 mainly by amending the requirement of sole control over the Article 26c from Directive 1999/93/EC being amended by "with a high level of confidence". The question if the secret signature generation key[194] can be kept under sole control at all has been debated[195], but currently the use of QSCD is the technically prevailing option and has seen legal acceptance in practice.

Roßnagel dismissed this model for Directive 1999/93/EC [410, 414] and for the Regulation 910/2014 he notes that this "reduces the security level"[196] [412, p. 3689] in order to gain the possibility of a legally acceptable remote signature generation by a third party.

Regulation 910/2014 remote signature allows a third party to execute the initial signing process on the signatory's qualified electronic signature creation device (QSCD) on the signatory's behalf, but remote signatures are new signatures, not a subsequent modification. Following Regulation 910/2014 the creation of a signature by an entrusted third party means the third party is given the right by delegation to generate a signature on certain data by operating the QSCD on the signatory's behalf. The entrusted QSCD is that of the signatory and it holds the signatory's secret signature generation key. Such a remote signature service generates a new signature on the submitted data using the signatory's secret key. Namely, the remotely signed data could also be previously unsigned. Hence, remote signatures in Regulation 910/2014 are neither technically nor operational a subsequent modification of an already signed document, but an initial signature.

Jandt in [270][a], as well as *Roßnagel* in [410, 412, 414] argue that this is a weakened notion — weaker due to the added "with a high level of confidence" [Art. 36 c Regulation 910/2014] compared to Directive 1999/93/EC.

The question of how "[...] sole control over the use of his electronic signature creation data [...]" [Regulation 910/2014] is exercised is not impeding that a remote signature is creating a **new** signature on **newly** submitted data. Hence, this thesis deduces that this change in Regulation 910/2014, while important, does not impact on the probative value for subsequent modifications of signed data.

[194] This is legally referred to as "electronic signature creation data" [Regulation 910/2014].

[195] See *Roßnagel* [412, 414] who sees the removal of sole control as a reduction of the security level; or *Kment* [286] who interprets this as Regulation 910/2014 offering more flexibility.

[196] Translated from the German original: "Herabsetzung des Sicherheitsniveaus" [412, p. 3689]; this view is re-stated in his work from 2016 [414].

[a] Translated from the German original: "Art. 36 Buchst. c Regulation 910/2014 fordert nur noch, dass der Siegelersteller Signaturerstellungsdaten — also den geheimen Schlüssel — 'mit einem hohen Maß an Vertrauen unter seiner alleinigen Kontrolle verwenden kann'. Dies ermöglicht die Erstellung des Siegels durch eine nicht berechtigte Person im Namen der juristischen Person." [270].

Further, it is out of scope of this thesis to analyse Regulation 910/2014's impact at this early stage of being in effect. Following *Roßnagel* this will probably only be possible after some court decisions [412, p. 3689]. For an analysis of the integrate-ability into German legislation the interested reader is directed to the analysis by *Roßnagel* [414] or *Jandt* [270]. Both additionally highlight the problem arising from the very high evidence with a legal presumption of authenticity that is assigned by Regulation 910/2014 to data with qualified electronic seals [270]. This thesis briefly checks the impact of the changes introduced by the German law by the eIDAS-Durchführungsgesetz (eIDASDG) [98], which made SigG and SigVO cease to have effect, and the newly introduced Vertrauensdienstegesetz (VDG) in Sec. 4.5.4.

In Sec. 4.4.8 it was found that remote signatures constitute a new fresh signature and are not obstructing the discussion of malleable signatures. It remains out of scope of this work to analyse how the 'sole-control-feature' found in Regulation 910/2014 must be achieved technically.

4.5. German signature law: Analysis of the impact of authorized modifications on the probative value in German signature law

In Sec. 4.2.2.2 this thesis found that the German legislation for the probative value of electronic documents point to electronic signature legislation. In order to be grounded in actual legislation, this thesis bases the legal analysis on concrete member state legislation, see Sec. 1.4.2 for further reasons. A detailed differentiation of Regulation 910/2014 and Directive 1999/93/EC regarding their differences has been presented in Sec. 4.4.6. For some background on the German laws that preceded the previous SigG[197] and the influence it had on electronic commerce in Germany see [288].

This section uses concrete member state legislation and legislative texts of Germany to identify what the functional requirements of a signature are (see Sec. 4.5.1) and that it gave the German courts the freedom to argue for other non-standardised signature generation protocols in a German case known as 'Container Signatur' (see Sec. 4.5.2). It then continues to identify what the notion of 'change' does mean, as it was not defined further in the EU texts (see Sec. 4.5.3). Moreover, the impact of the changes introduced following Regulation 910/2014, namely the German VDG and eIDASDG, are briefly looked at with respect to the probative value and the notion of a 'subsequent modification'. Finally, this section will analyse the permission of authorized changes in the light of legal delegation and the legal construct of blanket statements (see Sec. 4.5.5).

4.5.1. German signature law: Officially suggested interpretation of the required functionality of an electronic signature (BT-Drs 14-4987)

The reasons for the introduction of the changes to implement the statutory trust rules and the probative value introduced by qualified electronic signatures following the SigG and thus implementing the Directive 1999/93/EC into member state law[198] are given in BT-Drs 14-4987.

Note that this means that the reasons and explanations given in BT-Drs 14-4987 itself are not German law. Nevertheless, as officially suggested interpretation this is very useful for a teleological interpretation of the German signature legislation. BT-Drs 14-4987 [147, p.15-17] explains in non-technical language that in order to be treated equivalent to the paper form the electronic counterpart of a signature shall fulfil equivalent functions as the paper-based form of a signature. This thesis lists the functionalities here, as the thesis had considered them during the formulation of the technical requirements. They have not been added as their own requirement, but they are subsumed in the resulting requirements. Please see Sec. 7.10 which is a matrix between requirements and functions. The following seven functions are intended to be fulfilled by electronic signatures in Germany [147, 149]:

[197] The former German Signaturgesetz (SigG) and the accompanying Signaturverordnung (SigVO) codified the formerly binding Directive 1999/93/EC in terms of requirements for the German laws.

[198] Namely the law related to BT-Drs 14-4987 was the German 'Gesetz zur Anpassung der Formvorschriften des Privatrechts und anderer Vorschriften an den modernen Rechtsgeschäftsverkehr vom 13. Juli 2001'.

1. Conclusory function (in German ‘Abschlussfunktion’)
2. Archiving or integrity function (in German ‘Perpetuierungs- oder Integritätsfunktion’[199])
3. Identity function (in German ‘Identitätsfunktion’)
4. Authenticity function (in German ‘Echtheitsfunktion’)
5. Verification function (in German ‘Verifikationsfunktion’)
6. Evidential function (in German ‘Beweisfunktion’)
7. Warning function (in German ‘Warnfunktion’)

This thesis will not further elaborated here what each function comprises. The analysis if and how the functionalities can also be achieved by malleable signature schemes is given in the concluding chapter (see Chapter 17). Noteworthy, this high-level description of functionalities, even if it is only found in the reasons for the law that held changes to German legislation texts implementing Directive 1999/93/EC, has fortified the analysis result of this thesis that the German law sees electronic signatures in a technological neutral light (see Sec. 4.6.3.3).

4.5.2. German signature law: The legality of the ‘container signature’

The flexibility to analyse ‘new methods’ of electronic signatures that is introduced technological neutrality of the German signature law — by describing the requirements for electronic signatures by listing natural language descriptions of the intended functionality of a signature (see Sec. 4.5.1) — is reflected in a court ruling by the German Federal High Court of Justice (BGH). The BGH had to decide on a technical process that was used to sign a couple of documents[200]. The ruling allowed the used technical integrity mechanism to protect multiple documents in a group, a so-called “[c]ontainer” [81][201], with just one signature and then still treat each document from this container as a validly electronically signed document on verification[202] and the ruling stated that the technical mechanism correctly achieved the intention of the requirement of an electronic signature, i.e., they protect integrity and authenticity[203]. [81] However, note that this ruling was able to draw from an existing technical discussion on the used mechanism[204]. Further, the ruling was able to draw on the fact that the mechanism in question was already deployed as an established method to contact the court electronically[205].

4.5.3. German signature law: The notion of ‘subsequent change’ in German signature legislation as introduced by Directive 1999/93/EC

In Sec. 5.5.2 this thesis defines modification, as being based on identifying the notion of changes. In the SigG the “subsequent change” Directive 1999/93/EC is translated word by word into “nachträgliche Veränderung” [§ 2 paragraph 2.d SigG], with the same literal meaning.

§ 17 SigG describes the technical and procedural security from Annex II Directive 1999/93/EC. The German SigG uses the notion of “Verfälschungen signierter Daten” [§ 17 paragraph 1 SigG], that can be interpreted as meaning falsification of signed data.

[199] In [149] it is called ‘Perpetuierungsfunktion’; later it is referred to as “Perpetuierungs- oder Integritätsfunktion” [81, p. 6], which among other references also points to [147].

[200] BGH Beschluss vom 14. Mai 2013 – VI ZB 7/13 [81].

[201] In this ruling the multiple documents where referred to as a ‘container’ and technically it was presented that the container was depending on every single document’s content.

[202] “Die im EGVP-Verfahren — wie auch im Streitfall — eingesetzte qualifizierte Container-Signatur genügt den Anforderungen des § 130a ZPO” [81, para. 10].

[203] “[...] mit ihr werden Sinn und Zweck der qualifizierten Signatur — die Sicherstellung von Authentizität und Integrität des Dokuments — erreicht” [81, para. 10].

[204] For example, the mechanism referred to as ‘Container-Signatur’ was already discussed in 2009 in an Article published Datenschutz und Datensicherheit (DuD) by *Gennen* [203].

[205] Note that till 2016 the ’Elektronischen Gerichts- und Verwaltungspostfach (EGVP)’ was a way for electronic communication with courts; starting 2016, it will be subsequently phased out and replaced, e.g., by the ’besonderes elektronisches Anwaltspostfach’ following the ’Gesetz zur Förderung des elektronischen Rechtsverkehrs’ from 2013 [95].

German electronic signature legislation (SigG) requires detection of any subsequent modifications. SigG requires a detection, such that subsequent changes[a], forgeries or falsifications[b] are detected by the signature verification.

[a] In German: "eine nachträgliche Veränderung der Daten erkannt werden kann" [§ 2 paragraph 2.d SigG].

[b] In Article 17 and 19 SigG speaks of an 'undetected forgery' in German: "unbemerkte Fälschung" [§ 19 (4) SigG] or undetected falsification in German: "unbemerkte Verfälschung" [§ 19 (4) SigG].

Taking into account that electronic signatures shall help to generate evidence, such that the probative value is that of non-public documents executed by private party ('Urkunden') the German criminal code (StGB) discusses the falsification of documents in § 267:

> § 267 of StGB:
> "Whosoever for the purpose of deception in legal commerce produces a counterfeit document, falsifies a genuine document or uses a counterfeit or a falsified document, shall be liable to imprisonment not exceeding five years or a fine." [§ 267 (1) StGB]

The "purpose of deception" [StGB] is, following prevailing opinion, meant to include "any subsequent modification of the intellectual content of the genuine document to create the appearance that the declaration was made by the genuine document's issuer in the form that it has after the modification" [206] [434].

Subsequent modification must lead to "modification of the intellectual content" to be falsification under German criminal code. The prevailing opinion of interpretation of the German criminal code on falsification of documents wants the modification to lead to a change in the document's intellectual content. To decide that, the context of the processing of the document is required to identify if a change, e.g., a change of just the font of the text, would result in a "intellectual content" that is different from that of the genuine document. The German signature law in SigG and SigV do not forbid authorized modifications explicitly [§ 15 SigVO].

This led to Analysis Result 8 formulated in Sec. 5.5.1.

In Sec. 4.5.5 this thesis looks specifically at the legal implications when authorized changes are subsumed under a delegation. Next, in Sec. 4.5.4 the impact of the Regulation 910/2014 on German signature regulation and their handling of a subsequent modification is analysed.

4.5.4. German law: Impact of eIDAS to the treatment of subsequent modifications (especially VDG and eIDASDG)

In Germany, with Regulation 910/2014 coming into force the eIDAS-Durchführungsgesetz (eIDASDG) [98] and the Vertrauensdienste Gesetz (VDG) [151] have superseded legislation which has been analysed in the course of this thesis and by others, e.g. SigG or SigVO. This thesis has therefore analysed if eIDASDG or VDG introduce any new wording with respect to subsequent modifications.

Especially, as in the case of documents of official authority signed by natural persons there is no probative value assigned within Regulation 910/2014. Thus, for Germany the analysis of the ZPO a.F. is still valid. As discussed in Sec. 4.4.3, there is an important change with Regulation 910/2014 noted by *Roßnagel* [412]: The prescribed assignment of the probative value for seals found in Regulation 910/2014 — seals are signatures created by a legal person — would elevate its probative value to that of legal presumption[207] [412, p. 3691]. Thus, qualified electronic seals would give evidence with a legal presumption of authenticity or legal presumption[208] the same high probative value as documents of official authority signed by an advanced electronic signature. However, seals still come with the same requirements to detect any subsequent as electronic signatures do:

[206] Translated from German: "jede nachträgliche Veränderung des gedanklichen Inhalts einer echten Urkunde anzusehen, durch die der Anschein erweckt wird, als habe der Aussteller die Erklärung in der Form abgegeben, die sie durch die Verfälschung erlangt hat" aus *Schönke and Schröder* § 267, Rdnr. 64 [434].

[207] "Diese Vermutungen gehen hinsichtlich des Objekts und der Wirkung weit über den Anscheinsbeweis für qualifizierte Signaturen nach § 371a ZPO hinaus. Sie gelten nicht nur für das Beweismittel, sondern auch für Tatsachen. Sie sind nach § 292 ZPO nur durch den Beweis des Gegenteils widerlegbar, nicht bereits durch Erschütterung der Anscheinsgrundlage." [412, p. 3691].

[208] In German also translated with "gesetzliche Echheitsvermutung" [284].

> "Article 36
> Requirements for advanced electronic seals
> An **advanced electronic seal** shall meet the following requirements:
> (a) it is uniquely linked to the creator of the seal;
> (b) it is capable of identifying the creator of the seal;
> (c) it is created using electronic seal creation data that the creator of the seal can, with a high level of confidence under its control, use for electronic seal creation; and
> (d) it is linked to the data to which it relates in such a way that **any subsequent change** in the data is detectable."[209] [Article 36, Regulation 910/2014]

The "[...] qualified electronic seal based on a qualified certificate [...]" [Article 35 (3), Regulation 910/2014] is added as an alternative to the need for qualified electronic signatures in German legislative texts by the eIDASDG[210].

Neither VDG nor eIDASDG change passages in the legal texts that discuss subsequent modifications; the Regulation 910/2014 even heightens the assigned probative value for documents with qualified seals. Apart from taking away member-state legislation that is replaced by Regulation 910/2014 they do not introduce any new wording regarding the probative value other than those for the Regulation 910/2014 compared to Directive 1999/93/ECitself which were already analysed. Noteworthy is that Regulation 910/2014 explains that electronic documents with a qualified electronic seal, i.e., a signature by a legal person, are given a probative value higher than prima facie evidence (as analysed in Sec. 4.4.3 and mentioned by *Roßnagel* [412]). Still, a qualified electronic seal needs a qualified certificate to link to the legal person and needs to enable to be "[...] linked to the data to which it relates in such a way that any subsequent change in the data is detectable." [Article 36, Regulation 910/2014]. Hence, this thesis analysis of German legal texts done so far is still valid.

4.5.5. German law: The notion of 'blanket statements' with respect to MSS's delegation functionality

So far the analysis found that the prevailing opinion is that the German criminal code (StGB), namely § 267 (1) StGB, does not see it as a falsification if a subsequent modification is done in consent with the issuer of the document[211]. This is in line with jurisprudence; further German courts ruled that a signatory is allowed to leave certain blocks underspecified or empty, allowing them to be filled with information later. For example this underspecification has been used to explain what happens when a natural person issues a so called blank cheque as shown in Fig. 146. In legal parlance such underspecified declarations issued by the signatory is known as a blanket statement[212]. In German legislation blanket statements are a special form of a declaration of intent[213]. A blanket statement captures not only the contents of the declaration of intent, but also captures the signatories intention to leave it underspecified. Following the German § 172 BGB this can be achieved by a letter of authorization, that gives authority to the delegatee:

> "Letter of authorisation
> (1) If the principal has delivered a letter of authorisation to the agent and the agent presents it to a third party, this is equivalent to a separate notification of authorisation by the principal.
> (2) The power of agency remains effective until the letter of authorisation is returned to the principal or declared to be invalid." [§ 172 BGB]

As an electronic form for a declaration of intention is not prohibited, an electronic version of a blanket statement is possible as well.

[209] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[210] See the changes introduced in the German text where eIDASDG adds "[...] versehen sind mit [...] einer qualifizierten elektronischen Signatur, [...] oder [...] einem qualifizierten elektronischen Siegel." [Artikel 8,9,10, eIDASDG].

[211] *Schönke and Schröder* § 267, Rdnr. 67 [434].

[212] This is termed 'Blankett' in German or also "Blanketterklärung" [502].

[213] Declaration of intent is translated from the German 'Willenserklärung' according to Langenscheidt Recht Englisch [70].

Hence, a blanket statement can be created by the signatory when allowing authorized modifications to a signed document. The concept of a malleable signature allows the Sanitizer to modify only those blocks of data that the Sanitizer has been authorized to by the Signer. This has explicitly been termed[214] as the delegation of signing rights [see 18, 65, 224, 328] because the Sanitizer is a party different from the Signer, but acts on his behalf.

The concept of MSS can be seen as an encoding of the letter of authorization. With regards to the delegation this thesis generally sees that with the creation of a malleable signatures the Signer creates a "letter of authorisation", as found in § 172 BGB, for the Sanitizer to carry out authorized subsequent modifications. This of course requires the MSS to be technically suited to document and also to secure this authorization accordingly.

This thesis will show in Sec. 18.7 the concept of constructing an electronic blanket statement using malleable signature's modification policy ADM as the "letter of authorisation" found in § 172 BGB. This gives than the legal foundation to the blank signatures proposed by *Hanser and Slamanig* [224].

4.5.5.1. Signatory is legally liable for statements in a blanket statement

authorized modifications require signatory's consent to be no falsification and messages with such a signature are then a blanket statement. Modifications are only no falsification if they are done consented [*Schönke and Schröder*: § 267 StGB, Rdnr. 67 [434]]. And only if the signatory has consented to the generation of precisely this document by authorising modification, the signed document is a blanket statement following § 172 (2) BGB, as ruled by the German Federal Court of Justice [78].

Therefore, it would legally help if the Verifier of a signature can not only identify and verify the original Signer's signature of a blanket statement, as the identifiable originating legal entity, but also corroborate a proof to third parties, e.g., a court, that the Signer has acted explicitly with consent when generating the blanket statements.

Another important result for the accountability that follows from the applicability of § 172 BGB is that any specific information filled in later is attributed to the original signer of a blanket statement.

This was ruled in a case by the German Federal Court of Justice (BGH)[215]. If done in a consented way § 172 Abs. 2 BGB can be applied [See *Schramm* § 172 BGB, Rn. 17 [435]]; and the blanket statement forms a letter of authorization.

The signatory of a blanket statement is liable for any specific information filled into it later. Following German legislation § 172 Abs. 2 BGB the signatory of a blanket statement is liable for any specific information filled into the blanket statement later. From a BGH case follows that the signatory bears the full risk, even if the delegatee would not follow as intended at signature generation time.

Economically this risk could be acceptable, if the Signer delegates signing rights to a delegatee that can be held accountable for the subsequent information filled into the blanket statement. This way, the signatory might be able to file for compensation for damages from the delegatee later. There are ways to introduce additional limitations on the liability of a signature: *Fischer-Dieskau et al.* state that the German signature legislation allows the Signer to limit the value for which a signature generated shall be liability by having these limitations already stated in the public verification certificate [194].

[214] "The idea behind sanitizable signatures is that the signer delegates the signing rights of parts of the message to a designated party, the sanitizer." [65].

[215] BGH, ruling from 11.7.1963 - VII ZR 120/62 [78].

4.5.5.2. Delegation of signing rights in German law

The German Federal High Court of Justice (BGH) had to discuss the following case[216]: A lawyer participated in the electronic document exchange with the court. The lawyer's secretary was instructed by the lawyer to type a dictated pleading and digitally sign the resulting electronic document containing the pleading before sending it to court in electronic form. The signature was generated by the lawyer's personal smart card and the lawyer's PIN had been shared with the secretary. The electronic pleading was signed and sent electronically in due time. However, later the court wrote a letter that the signature validation failed. The lawyer raised a plea, to prolong the deadline, and argued as follows:

If the signature validation failure would have been immediately indicated, the secretary would have faxed it within deadline instead. The court overturned his plea arguing that the secretary is not an authorized proxy, i.e., could not act on behalf of the lawyer in front of the court. As the secretary applied the faulty electronic signature on the electronic version of the written pleading, but because the secretary was never legally allowed to be a delegatee of this task, the court does not need to hear the secretary. In other words, the courts decision is solely based on the fact that the court never had to consider the secretaries input regardless of the validity of the lawyer's signature. Hence, the court ruled that the written pleading handed in by the secretary did not count.

Previous to Regulation 910/2014, the German SigG and SigVO followed the Directive 1999/93/EC and required that for qualified electronic signatures the secret signing key is linked to one single natural person [§ 5 SigG]. The secret signing key is kept secret in a secure signature-creation device (SSCD) and its usage needs to be unlocked by the associated natural person using a PIN, password or even biometric identification to generate a signature. This is how the binding between the legal signatory and the technical Signer is technically supported.

What the accessibility of the signing key legally means to enforce the link between the cryptographic material and the natural person is getting clear when looking at some discussions: The German notaries explicitly state in DNotZ 2006, 561 that solutions that require to give or to deposit the secure signature-creation device (SSCD) at another notary are not able to produce legally binding signatures [86] and that PIN and secure signature-creation device (SSCD) are not allowed to be shared with staff members or third parties[217]. Previous to Regulation 910/2014 *Roßnagel* dismisses that third parties can generate legally binding signatures on behalf of the signatory when the signatory is providing them with the signatories secure signature-creation device (SSCD) and the knowledge, e.g., PIN, needed to operate it [410]. *Roßnagel* sees a violation of the requirement that the signatory can still keep the means under his sole control[218] [410]. Hence, *Roßnagel* concludes that statutory evidentiary presumption can no longer be awarded in this case [410].

With the advent of Regulation 910/2014 remote signature generation became legally more grounded. The possibility for a remote signature generation has been analysed in Sec. 4.4.8.

Remote or delegated signature generation is regarded as generating a new signature, which is different to the subsequent modification of already signed data done for blanket statements or by malleable signature schemes. As analysed in Sec. 4.4.8, the question of how "sole control over the use of his electronic signature creation data" [Regulation 910/2014] is exercised is not impeding that a remote signature is creating a new signature on newly submitted unsigned data. Hence, Regulation 910/2014 allowing such deferred signature generation does not impact on the analysis of the probative value for subsequently modified signed data. With a malleable signature scheme the delegated party, the Sanitizer can only take subsequent authorized actions if the data is already verifiably signed by the Signer.

Technically possible delegation is separate from their legal recognition and vice versa. A qualified signature-creation device (QSCD) — as well as a secure signature-creation device (SSCD) — bound to the signatory by a PIN technically allows to delegate the signing procedure to the delegatee by sharing the knowledge of the signatory's PIN to unlock the secret signing key and giving the delegatee access to the signatory's qualified signature-creation device (QSCD). By that, the signatory allows the delegatee to sign any document on her or his behalf. Particularly this sharing allows the delegatee to

[216] BGH from 21.12.2010, Az.: VI ZB 28/10 [80].

[217] See Richtlinienempfehlungen der Bundesnotarkammer IV.2. [87].

[218] § 2 Nr. 2 c) SigG.

create many, not just one, blanket statement and gives the delegatee the complete freedom to construct each document's signed content. In the discussed ruling[a], the German Federal High Court of Justice (BGH) decided that in this case a blank signature does not fulfil the signatory's legal obligation[b] to finally check and approve the documents correct content [80]. Vice versa, just because there was no legally allowed delegation in a case, does not mean that it has has to be technically prevented.

This thesis offers in Sec. 18.7 a discussion of how to technically build blanket statements using malleable signatures.

[a] BGH from 21.12.2010, Az.: VI ZB 28/10 [80].

[b] In the ruling from Footnote *a* the signatory was the lawyer.

4.6. Analysis results of legal definitions regarding integrity and a high probative value

To conclude, this section compiles in condensed and summarised form the intermediate findings and results obtained from the thorough analysis of the legal acts' texts. The results, together with the analysis of computer science texts followed in Chapter 5 forms the basis of this thesis definition of Integrity presented in Chapter 6.

4.6.1. Analysis Result 1: Legal texts acknowledge the importance of integrity

The need for integrity protection is clearly stated in legal texts and legal discussions. This thesis suggests in Sec. 4.3.1 a textual interpretation to group integrity alongside the goals of confidentiality and availability. These are the three classic goals of IT security [125, 438]. The reworded Article 13a(3) of Directive 2009/140/EC [180] would then read:

> "Member States shall ensure that undertakings providing public communications networks or publicly available electronic communications services notify the competent national regulatory authority of a breach of ~~*security*~~ *confidentiality* or loss of integrity that has had a significant impact on the operation of networks or services."[219] [180] .

The German Supreme Constitutional Court (BVerfG), Germany's highest court, identified the new basic right of confidentiality and integrity of information technology systems [100]. Integrity is declared as an important and separate protection goal. However, different to this thesis the basic right constituted in 2008 has the scope of protecting the IT system as a whole.

Both EU general view on IT Security, after the technical refinement of Article 13, and the German basic right, highlight integrity as different from confidentiality. Thus, EU and also German legislation acknowledge integrity as a protection goal of its own and put integrity in line with other known IT security goals.

4.6.2. Analysis Result 2: Legal acts do not explicitly forbid authorized modifications

Some general definitions found in European acts related with ENISA provide no clear statement: On the one hand, ENISA defines that integrity-protected data has to remain "unchanged" [179, Article 4 lit. f], which can be interpreted grammatically to forbid any form of subsequent change including any authorized and benign modifications. On the other hand, the same ENISA regulation defines what integrity should protect against when defining the security goals in the definition of "Integrity of Networks and Services". Herein, the goal of integrity for network and information systems means to resist "[...] unlawful or malicious actions that comprise [...] integrity [...]" [179][220]. Section 4.3.1 put this into context to aid the grammatical interpretation. From context like Article 13a(2) of [180] this thesis deducts that malicious activity that attacks the availability is the central concern of ENISA's definitions. Taking this into account, ENISA's grammatically different definitions can be harmonised and more weight is

[219] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[220] Article 4 lit. c of the Regulation (EC) No 460/2004 [179].

given to the above definition from the ENISA [179]. This positions ENISA's integrity protection as a technical mechanism to resist "malicious actions" [179]. Hence, this thesis concludes that ENISA implicitly tolerates lawful or benign actions like authorized modifications unless they would comprise the integrity.

Some legal texts are implicitly allowing authorized subsequent modifications: EU Regulation 2016/679 (GDPR) on data protection (see Sec. 4.3.3 for an analysis) and US FISMA or US HIPAA (see Sec. 4.3.4 for an analysis). The latter explicitly identifies only "improper alteration" [484] as a violation that integrity protection mechanisms should detect or prohibit. Thus, like the GDPR and US FISMA, US HIPAA implicitly tolerates authorized subsequent modifications. In that light, ENISA's re-interpreted integrity notion is also in line.

Thus, this thesis concludes that none of the legal texts analysed explicitly forbids authorized modifications. Moreover, the analysis yields that legal acts raise no objections against distinguishing modifications further into authorized and unauthorized modifications. Furthermore, legal acts do not explicitly state the impact of detected, but authorized modifications on the integrity itself; neither negative nor positive impacts are mentioned.

4.6.3. Analysis Result 3: European and German legal texts discuss the rules for probative value and statutory evidentiary presumption for digital/electronic documents; and they point to electronic signature legislation which describe the requirements for qualified/advanced electronic signatures in technology-neutral language

As the EU legislations do not directly assign probative value[221] this thesis followed the German Code of Civil Procedure (ZPO). Using one precise legal regime allows to analyse in detail the legal concept of probative value and statutory evidentiary presumption for documents that exist in computer systems. The EU legislations name these documents *electronic documents*. Further, the United Kingdom's (UK) legislation is briefly analysed in this section; and it was found that the UK has not introduced direct legislation regarding the probative value for the electronic document as a whole.

4.6.3.1. Analysis Result 3.1: If signed with a qualified electronic signature, the digital/electronic documents gets awarded with a high probative value and statutory evidentiary presumption

The analysis using grammatical interpretation of the German ZPO yields, that the general rules for the probative value of non-public documents executed by private party apply to their electronic form. In German law this establishes a full proof that the person who signed the non-public document executed by private party issued the declarations that are contained in the signed non-public document executed by private party. This is referred to as statutory evidentiary presumption. However, this proof is only established if the authenticity of the electronic form can be judged according to the Regulation 910/2014[222]. To gain statutory evidentiary presumption the ZPO sets forth that the highest requirements for the electronic signature must be fulfilled. On the European legal level the electronic signatures meeting these are called *qualified electronic signatures* in Regulation 910/2014[223]. The former German SigG implemented the now superseded EU Electronic Signature Directive 1999/93/EC. Following SigG they are called a "qualified electronic signature based on a qualified certificate" [SigG]. They must be issued by a secure signature-creation device (SSCD) [§ 2 (3) SigG][224]. The updated Regulation 910/2014 names this a *qualified signature creation device (QSCD)*. Due to the technical importance the need for an QSCD is separately stated as Analysis Result 6.

[221] See for example Recital 49 of Regulation 910/2014stating "[...] it is for national law to define the legal effect of electronic signatures, except for the requirements provided for in this Regulation according to which a qualified electronic signature should have the equivalent legal effect of a handwritten signature." [Recital 49 Regulation 910/2014].

[222] It formerly pointed to the German Electronic Signature Act (SigG).

[223] Formerly Directive 1999/93/EC called them 'advanced electronic signatures'

[224] In German "qualifizierte elektronische Signaturen [...] die a) auf einem [...] qualifizierten Zertifikat beruhen und b) mit einer sicheren Signaturerstellungseinheit erzeugt werden [...]" [§ 2 (3) SigG].

To conclude, non-public documents executed by private party with a valid qualified electronic signature following the highest standard described by the Regulation 910/2014[225] have a high probative value and they establish the full proof that the contents

- are issued by the Signer[226] [227],
- are a declaration having been made by the Signer [226], and
- are presumed to be authentic unless "facts giving rise to serious doubts" [ZPO].

For non-public documents executed by private party and documents of official authority, a high probative value and the statutory evidentiary presumption as prima facie evidence can be awarded.

4.6.3.2. Analysis Result 3.2: Rules for probative value point to electronic signature legislation, which describe the requirements for a qualified electronic signature

As the high probative value and statutory evidentiary presumption will only be granted if the electronic document has a valid qualified electronic signature, EU member state's regulation for the probative value pointed to electronic signatures legislation, or is directly influenced by EU Regulations concerning electronic signatures. Thus, the technical and organisational requirements were formerly set forth by EU Directive 1999/93/EC — indirectly as it was transposed into member state laws like SigG and SigVO — and now directly by the Regulation 910/2014. See Sec. 4.6.6 for details of the legal effect. Regulation 910/2014, which replaced Directive 1999/93/EC, has the same goals, as shown in Subsections 4.4.3 and 4.4.6.

The following list summarises the most important requirements imposed by the highest technical requirements that are decreed for so-called *qualified electronic signatures*:

- Section 1(c) of the requirements listed in Annex II of the Regulation 910/2014 clearly states that "[...] the electronic signature is reliably protected against forgery using currently available technology." [Annex II 1(c), Regulation 910/2014].[228]
- A signature must offer protection against undetected "subsequent changes" [Regulation 910/2014].[229] The EU legislation on electronic signatures postulates — technologically neutral — that changes of signed data that happen after the signature has been created must be detectable.
- A signature must link the statement made in the signed text and the signatory (Signer). For this link the signatures must offer two properties:
  (1) A signature must provide integrity protection for the data that it covers, i.e. the signature must be "[...] linked to the data signed therewith in such a manner that any subsequent change of the data is detectable." [Article 26 (d), Regulation 910/2014].[230]
  (2) A signature must provide authentication of origin (authenticity), i.e., "uniquely linked" [Article 26 (a), Regulation 910/2014] and "capable of identifying the signatory" [Article 26 (b), Regulation 910/2014].[231]
- A signature must be created with a qualified electronic signature creation device (QSCD) that reliably
  (1) assures the confidentiality of the secrets necessary to generate the signature[232] [Annex II 1(a), Regulation 910/2014] and

[225] As previously also under Directive 1999/93/EC.

[226] This thesis identified the role of the Signer to have access to the secret signing key, so the Signer as correspondent to the legal "holder of the signature key" [ZPO a.F.] or the legal "person responsible" [ZPO n.F.] denoted also as the legal "signatory" [ZPO].

[227] §§ 415, 417 and 418 ZPO, § 371a Paragraph 1 ZPO.

[228] Same as in Section 1(b) of former Directive 1999/93/EC [Directive 1999/93/EC].

[229] Was the same in the former Directive 1999/93/EC [Directive 1999/93/EC].

[230] Slightly different worded in the former Directive 1999/93/EC: "[...] linked to the data to which it relates in such a manner that any subsequent change of the data is detectable [...]" [Directive 1999/93/EC].

[231] Same as in the former Directive 1999/93/EC [Directive 1999/93/EC].

[232] The secret signing key $\mathsf{sk}_{\mathsf{sig}}$ is legally termed "electronic signature creation data used for electronic signature creation" [Regulation 910/2014].

(2) allows the legitimate signatory to guard the secrets necessary to generate the signature "against use by others" [Annex II 1(d), Regulation 910/2014] (see Analysis Result 6).[233]

4.6.3.3. Analysis Result 3.3: Electronic signature legislation texts describe required functionality in technology-neutral language

Technically, digital signature mechanisms based on asymmetric cryptography (i.e., a pair of keys with a public and a secret key) together with a form of public key infrastructure allow to fulfil the legal requirements of *advanced electronic signatures which are based on a qualified certificate and which are created by a secure-signature-creation device* [Directive 1999/93/EC] or *qualified electronic signatures* [Art. 3 Regulation 910/2014].[234] *Laborde* points out, that the directive [175] "implicitly" [303] endorses cryptographic digital signatures, such as RSA-PSS, for *advanced* electronic signatures by "establishing requirements that, so far, can only be fulfilled by using digital signatures" [303]. However, this thesis finds that Regulation 910/2014 — as well as Directive 1999/93/EC — are formulating the requirements in a language which does neither contain names nor pointers to precise technologies.

Regulation 910/2014 especially mentions that additional legal acts need to specify details. In German law it is the ZPO that assigns the probative value. As depicted in Fig. 26 by the arrows, the interconnection is as follows: Before the Regulation 910/2014 was in effect the ZPO a.F. pointed to SigG, which in turn points to SigVO, which finally contains a pointer to a technical document maintained by a German government body issuing technical specifications, the German Federal Office for Information Security (BSI) [84]. With Regulation 910/2014 in effect the ZPO n.F. points directly to Regulation 910/2014 (SigG and SigVO have ceased to have effect). Note, unchanged after Regulation 910/2014 is that it is the ZPO that assigns the probative value. Also, still based on the German SigVO a technical document maintained by a German government body issues yearly the technical specifications of algorithms and their security parameters to be considered secure: The catalogue of algorithms issued by the Bundesamt für Sicherheit in der Informationstechnik (BSI) [84, 85]. The VDG[235] carries this over and § 2 (2) VDG specifies that the security assessment of algorithms and their parameters[236] is still with the BSI.

In general, those technical specifications, like [85, 354], are subordinate to a law. The law itself is termed technology-neutral[237]; this allows that the government can — in theory — make adjustments to follow important technological and cryptographic advances without changing the law's content by changing the technical specifications; or the courts can on a case by case basis determine the probative value for certain electronic signatures based upon their interpretation — potentially guided by hearing technical expert witnesses — of the technologically neutral requirements[238]. In the area of EU member state law there are also good examples: Not in German law, but it uses the following technologically neutral formulation: Electronic signatures must be functionally equivalent to handwritten ones [147, 149] and it also formulates seven functionalities in technology-neutral language (see Sec. 4.5.1). Based on these the German federal high court of justice (BGH) was able to legally approve applications of algorithms listed in the BSI issued yearly catalogue [84, 85] even if they did operate slightly different from existing

[233] The Regulation 910/2014 further enables that the "[generating] or managing electronic signature creation data on behalf of the signatory may only be done by a qualified trust service provider." [Annex II 3(, Regulation 910/2014]. By this the so-called remote signatures are enabled in Regulation 910/2014. This was different in the former Directive 1999/93/EC which stated "[...] the signatory can maintain under his sole control[...] " [Article 2, 2.(d), Directive 1999/93/EC], see Sec. 4.4.7 for a detailed grammatical comparison. Still, Annex III contains the same statement regarding the SSCD ensuring that "[...] the signature-creation-data used for signature generation can be reliably protected by the legitimate signatory against the use of others." [Annex III Directive 1999/93/EC].

[234] One should note that from a legal perspective, also non-cryptographic solutions such as scanned handwritten signatures are considered to be a form of *electronic* signatures, namely just electronic ones, but are not strong enough to give statutory evidentiary presumption.

[235] The VDG

[236] In German: "[...] (2) Von der Aufgabenzuweisung an die Bundesnetzagentur unberührt bleiben die Aufgaben des Bundesamtes für Sicherheit in der Informationstechnik nach dem BSI-Gesetz und nach weiteren Fachgesetzen, insbesondere [...] 2. die Bewertung von Algorithmen und zugehörigen Parametern [...]" [§ 2 (2) VDG].

[237] Recital 26 of Regulation 910/2014 demands that "[b]ecause of the pace of technological change, this Regulation should adopt an approach which is open to innovation." [Recital 26, Regulation 910/2014].

[238] Recital 27 of Regulation 910/2014 demands in this respect that "[t]his Regulation should be technology-neutral. The legal effects it grants should be achievable by any technical means provided that the requirements of this Regulation are met." [Recital 27, Regulation 910/2014].

legally recognised signature schemes. Namely, the algorithm in question did not directly sign a single document but several documents together and the BGH ruled that this offers the same protection for each individual document as if it would have been signed individually[239] (see Sec. 4.5.2 for a more detailed discussion of this ruling).

4.6.4. Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected

The EU signature legislation requires producing electronic signatures, that would give the signed document a high probative value, such that the signature is "linked to the data signed therewith in such a way that any subsequent change in the data is detectable" [Article 26 d Regulation 910/2014][Article 2 2.d Directive 1999/93/EC]. The analysed legislation does specify this as the minimum level of detection, authorized modifications as defined in this thesis (see Definition 105) are no exception. Regarding unforgeability, Section (1b) of the requirements, set forth by the EU Directive 1999/93/EC, states that a signature must not be forged once created [Directive 1999/93/EC].

Legally, statutory evidentiary presumption is removed once a subsequent modification, authorized or not, has been detected by the digital signature verification process. Only if any subsequent modification can be detected the signed data can be awarded a higher probative value compared to unsigned data. This sets forth the minimum required detection, but does not prohibit a more fine-grained detection of modifications. In other words, the directive does not specify to what additional level of detail a subsequent modification can be detected but that it must not remain unrecognisably hidden. All EU definitions are stringent in that they strictly forbid undetectable subsequent modifications of signed data.

The analysis yields that a legally compliant electronic signature scheme must offer protection against any undetected "subsequent change" [175, 182]. EU signature legislation — current Directive 1999/93/EC as well as future Regulation 910/2014 — does not treat authorized modification different from any other modification. This is in line with more general legal texts (see Analysis Result 2). Both legislative acts define as the bare minimum detection the detectability of any change to the electronic data that was done after it was signed. Both, do not forbid authorized changes to be detected as a separate class of detections, as long as they are detected.

4.6.5. Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory

Accountability as a notion is not explicitly stated neither in Regulation 910/2014 [182] nor in Directive 1999/93/EC [175]. Fundamental to the technical goals of origin authentication, accountability or non-repudiation is the statement that a signature must not be forged once created given, i.e., unforgeability is required, in Section (1b) Directive 1999/93/EC. Additionally to unforgeability the Directive 1999/93/EC requires a technically strong linkage between the signatory as a legal entity and the corresponding public verification keys.

Article 3 Regulation 910/2014 defines authentication of origin:

> " 'authentication' means an electronic process that enables the electronic identification of a natural or legal person, or the origin and integrity of data in electronic form to be confirmed;" [Regulation 910/2014].

Technically this can be implemented by certified public verification keys where the certificate is issued by a trusted certificate authorities and identifies the signatory, e.g. by stating the signatory's real name, birthday and address. This got defined as a *qualified public key certificate* as stated in Definition 32.[240]

[239] In the German literature often referred to as 'container signature'.

[240] This thesis notes that there is a more fundamental debate if the linkage can be achieved; compare *Sorge* that discusses that "Mason [[330], pp. 119–120] states the unique link of a signature to a person was impossible to achieve" [451] citing *Mason* [330]. However this thesis follows current legislation and technical argumentation that the technical evidence generated by a cryptographically secure CDSS with a legally certified and correctly working qualified public key certificate issuer attests, to the best procedural and technical effort, that the signatory is the origin of the signed document's contents.

Regarding the notion of signatory, in Directive 1999/93/EC the "signatory means a person who holds a signature-creation device and acts either on his own behalf or on behalf of the natural or legal person or entity he represents" [175]. In Regulation 910/2014 legal entities might use electronic seals, thus " 'signatory' means a natural person who creates an electronic signature" [Regulation 910/2014]. Capturing both, this role is denoted *signatory* as defined in Definition 19 and Signer refers to the entity who can technically create valid signatures due to control of the required secrets, as defined in Definition 18 in this thesis.

The Directive 1999/93/EC's recital 21 states that the provision of evidence for the origin of messages is important:

> "In order to contribute to the general acceptance of electronic authentication methods it has to be ensured that electronic signatures can be used as evidence in legal proceedings in all Member States" [recital 21 Directive 1999/93/EC]

Directive 1999/93/EC sets forth two requirements in the direction of origin authentication:

- Directive 1999/93/EC, as well as Regulation 910/2014, states that "it is uniquely linked to the signatory" [Article 2 a Directive 1999/93/EC][Article 26 a Regulation 910/2014] and "is capable of identifying the signatory" [Article 2 b Directive 1999/93/EC][Article 26 b Regulation 910/2014].
- In the recommendations for a secure verification process, Annex IV hints that a compliant scheme is capable to achieve that "the result of verification and the signatory's identity are correctly displayed" [175] and that "the use of a pseudonym is clearly indicated" [175].

Regulation 910/2014 follows Directive 1999/93/EC here, and Article 26 a and b is the same as Article 2 a and b of Directive 1999/93/EC.

To conclude, this view of authentication of origin is in line with what computer science describe as authentication of origin, see the Definition 23 stated in Sec. 2.10.2 and the analysis offered in Sec. 5.2.

4.6.6. Analysis Result 6: Qualified electronic signatures on electronic documents when computed by qualified electronic signature creation device (QSCD) award the same probative value that handwritten signatures award to paper documents following member state rules

Both Directive 1999/93/EC and Regulation 910/2014 award the digitally signed electronic document the same legal probative value that a handwritten signature would have awarded for a paper document if the signature verification outputs that the Signer has signed it and if the generation of the signature was done according to the rules laid down in the respective legal texts.

> "Member States shall ensure that **advanced electronic signatures** which are **based on a qualified certificate** and which are **created by a secure-signature-creation device**:
>
> (a) satisfy the legal requirements of a signature in relation to data in electronic form **in the same manner as** a handwritten signature satisfies those requirements in relation to paper-based data; and
>
> (b) are **admissible as evidence** in legal proceedings."[241] [Art. 5 (1) Directive 1999/93/EC]

In the Directive 1999/93/EC [175] those rules are given by the three requirements from Art. 5 (1). Namely, the electronic signature must be (1) "advanced" [175] and (2) "based on a qualified certificate" [175] and (3) computed "by a secure-signature-creation device" [175]. In the Regulation 910/2014 [182] the wording for those three requirements is slightly changed and names it a (1) "qualified electronic signatures" [182]. Qualified electronic signatures must also be (2) "based on a qualified certificate" [182] and must be (3) computed "by a qualified electronic signature creation device" [182].

> " **Legal effects of electronic signatures**
>
> [...]
>
> 2. A qualified electronic signature shall have the equivalent legal effect of a handwritten signature. " [Art. 25 Regulation 910/2014]

[241] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

Further, Regulation 910/2014 contains a statement that the protection's strength shall be at the height of the state of the art. Namely, it demands that the "signature is reliably protected against forgery using currently available technology" [Annex II para. 1.(c) Regulation 910/2014]. The complete analysis can be found in Sec. 4.4.7.

Henceforth, QSCD will be used to refer to the hardware, software and procedures that fulfil the requirements of Annex II Regulation 910/2014. The analysis has shown that the effects of the differences between a QSCD and an SSCD are marginal and make no difference for the probative value in the light of authorized subsequent modifications.

It is important to note that the actual probative value for a signed document was neither codified in Directive 1999/93/EC nor in Regulation 910/2014. As analysed (see Sec. 4.1) the — in this respect detailed — German Code of Civil Procedure (ZPO) awards the signed document at least a high probative value of prima facie evidence. Following *Roßnagel* this remains the same under the Regulation 910/2014 from June 2016 onwards[242] [412, 413]. Recital (49) Regulation 910/2014 acknowledges that, when fulfilling the highest standards of a "qualified electronic signature", the equivalent legal effects of a handwritten signature shall be awarded. It explicitly leaves the legal effects to the probative value open to member state regulation by stating:

> "(49) [...] However, it is for national law to define the legal effect of electronic signatures, except for the requirements provided for in this Regulation according to which a qualified electronic signature should have the equivalent legal effect of a handwritten signature." [Recital 49 Regulation 910/2014]

Hence, the German member state's regulation regarding the probative value of documents as analysed still applies[243].

4.6.7. Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights

A valid qualified electronic signature following the highest standard described by the EU Electronic Signature Directive [175] establishes the full proof that the contents are issued by the Signer and are a declaration having been made by the Signer. Details have been discussed in Sec. 4.6.3. A malleable signature scheme allows the Sanitizer to modify a block of signed contents as authorized by the Signer. This can be seen as the delegation of signing rights [18, 65, 328], because the Sanitizer is a party different than the Signer and is able to generate valid signatures attributed to the Signer on messages. This Sanitizer is then the delegatee who could modify a block, i.e., like in a blank cheque, while a Verifier would still yield that the Signer, here the delegator, signed the contents. Legally these are blanket statements, sometimes referred to as blanket declarations in German 'Blanketterklärung'. If done in a consented way, then any specific information filled in later is attributed to the original Signer of a blanket statements.[244] Therefore, it would legally help if the Verifier of a signature can firstly identify the legal signatory as the originating entity by verification of the original Signer's signature on a blanket statement; secondly the Verifier shall also be able to corroborate a proof that convinces third parties, e.g. a court, that the Signer has explicitly and with consent endorsed the subsequent modifications.

[242] In detail *Roßnagel* sees no contradiction of Regulation 910/2014 and the rules of evidence given in § 371a para. I and para. III ZPO, because Article 25 Regulation 910/2014 does not contain any rules of probative value for qualified electronic signatures; originally in German: "Kein Widerspruch zur VO [Regulation 910/2014] besteht auch hinsichtlich der Beweisregelungen in § 371a Absatz I und Absatz III ZPO, weil Artikel 25 Regulation 910/2014 keine Beweisregelung für qualifizierte Signaturen enthält." [412, p. 3691].

[243] For more details see *Roßnagel* [413].

[244] When consented, § 172 Abs. 2 of the German Civil Code (BGB) can be applied [see *Schramm* § 172 BGB, Rn. 17, 435]; this was also ruled by the German Federal Court of Justice (BGH) 11.7.1963 - VII ZR 120/62 [78].

5 Analysis of existing technical integrity and authenticity definitions with respect to authorized modifications

Overview of Chapter 5

Chapter 5 discusses existing computer science definitions of the terms *Data Integrity* and *Data Origin Authentication* with the focus on their statements towards authorized subsequent modifications. The analysis includes the work of *Biba*, *Clark and Wilson*, and standards *TCSEC*, *ITU X.800*, *ISO 27000/27001*, *ISO 7498-2*, *ISO 13888-1*, *ISO 10181-6*, and *RFC 4949*.[245] The standards were selected based on their adoption and widespread usage as a reference, and also due to their visibility towards governmental bodies, e.g. [186, 471]. Furthermore, the security textbooks and recent work from *Bishop*, *Gollmann*, and *Stallings* are analysed. Finally, an influential cryptographic security notion of existential forgery under chosen message attack, known as EF-CMA and its countermeasure as EUF-CMA, given by *Goldwasser, Micali, and Rivest* is analysed in the light of authorized subsequent modifications.

- Section 5.1 analyses different *technical* definitions of the term integrity to identify if they include the action of authorized subsequent modification, which is central to malleable signatures (see definition in Chapter 3).
- Section 5.2 describes what definitions of non-repudiation and authentication of origin *technically* offer towards the goal of authenticity for information that has undergone authorized modifications. See Chapter 4 for the legal background.
- Section 5.3 analyses the *technical* definitions of classic digital signature schemes regarding their treatment of authorized subsequent modifications.
- Section 5.4 shortly revisits definitions from the MSS literature to highlight the absence or shortcomings of existing notions that capture the integrity protection offered by MSS.
- Section 5.5 presents the results of Chapter 5.

5.1. Authorized modifications in existing technical definitions regarding integrity

This section analyses existing definitions for the terms of "Data Integrity" in computer science regarding authorized modifications: the *Biba*-model from 1977, *Clark and Wilson* from 1987, the Trusted Computer System Evaluation Criteria (*TCSEC*) from 1983. Moreover, the computer science textbooks definitions from *Stallings*, *Gollmann*, and *Bishop*. Furthermore, it will analyse definitions by the following standards *ISO 27000/27001*, *ISO 7498-2*, *ITU X.800*, *ISO 10181-6*, and *RFC 4949*.

[245] Note, for brevity this thesis does not reference standards as ISO/IEC XXXX but only as ISO XXXX.

5.1.1. Integrity considerations of secure computer systems by *Biba*

Today, the model from 1977 described in [47] is known under his author's name *Biba*. The *Biba*-model defined the following notion of integrity.

Definition 67 : Integrity of a subsystem by *Biba*

> *"We consider a subsystem to posses the property of integrity if it can be trusted to adhere to a well-defined code of behaviour. No a priori statement as to the properties of this behaviour are relevant." [47, p. 9].*

Biba assumes the "[...] subsystem has been initially certified (by some external agency) to perform properly." [47, p. 9]. While the above given definition of integrity talks about the integrity of the subsystem, the subsystem includes additionally all data handled by this subsystem. "The integrity of information is maintained by guaranteeing that **only proper modifications** are made."[246] [47, p. 14].

Regarding the type of modifications that *Biba* considers to fall into "[...] the notion of "proper" modification [...]" [47, p. 14], he clearly states that those "[...] policies are dependent on problem specific protection requirements." [47, p. 14]. *Biba* explicitly explains and lists threats to the integrity. The system's integrity is compromised if "[...] a system can be improperly persuaded (forced) to change its behaviour [...]" [47, p. 10]. *Biba* defines access control policies. The following strict policy is often assumed when referring to the *Biba*-model. Under the strict policy "[...] objects may not be directly modified by subject possessing insufficient privilege." [47, p. 30]. If the subject has sufficient privilege it can however modify the object without lowering the object's integrity. *Biba* "[...] assumes that the modifications made by an authorized subject are all at the explicit direction of a non-malicious program." [47, p. 30].

Regarding modifications, *Biba* generally discusses that to keep a system's integrity, only ""proper" modification" [47, p. 14] shall be happening. Following the strict security policy of *Biba*, the system prohibits access to object's if such an access would need to decrease the integrity of the subject or the object. Hence, integrity protection implementing *Biba*'s strict policy would prevent integrity violations by enforcing proper access control.

The "Low-Water Mark Policy" [47, p. 23] discussed by *Biba* needed to adapt the integrity classifications of either subjects or objects after certain access had happened. This would not prevent loss of integrity on objects, i.e., improper access to an object would lower the object's integrity level to that of the subject after each access. As *Biba* notes, this policy could be used as "Audit Policy" [47, p. 26] that offers detection of "a "level of corruption"" [47, p. 26].

***Biba* allows explicitly authorized modifications to not violate the integrity.** *Biba* defines mechanisms for prevention and for detection of integrity violations in [47]. The interpretation of the subject classification and their use to authorize access (including access for modification) is that integrity is not lost if the subject has the same or a higher integrity classification.

Biba describes to use the "Low-Water Mark Policy" [47, p. 23] as an "Audit Policy" [47, p. 26] to offer detection of "a "level of corruption"" [47, p. 26]. This would result in the ability to detect unauthorized subsequent modifications. Regarding the detection of subsequent authorized modifications, *Biba* makes no explicit statements.

5.1.2. Data integrity by the Trusted Computer System Evaluation Criteria (TCSEC)

In 1983 the US Department Of Defense set out to define more technical properties that must be present in a trusted computer system. The Trusted Computer System Evaluation Criteria (TCSEC) [308] from 1985 defines, apart from system integrity and label integrity, Data Integrity as follows:

[246] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

Definition 68 : Data Integrity by TCSEC

> "'Data Integrity – *The state that exists when computerized data is the same as that in the source documents and has not been exposed to* ***accidental or malicious alteration*** *or destruction."*[247] [248] *[308]*

TCSEC touches the notion of external consistency[249] when stating that "[...] computerized data is the same as that in the source documents [...]" [308].

TCSEC implicitly tolerates authorized modifications. TCSEC does not explicitly consider benign or non-malicious or non-accidental modifications as threats to the integrity. Regarding modifications, TCSEC explicitly lists only "accidental or malicious alteration" [308] as modifications which are a threat to the state of integrity. Hence, TCSEC implicitly tolerates authorized modifications that do not violate the integrity.

Regarding the detection of subsequent authorized modifications, TCSEC makes no explicit statements.

5.1.3. Data integrity by *Clark* and *Wilson*

In 1987 *Clark and Wilson* saw that in the commercial world data must be manipulated, but not just arbitrarily. They identified that data cannot always be routed through a "security officer or trusted process" [125]. So, they introduced "well-formed transactions" [125] instead, which allow certain conditions to be fulfilled during data modification, i.e. audit logs. *Clark and Wilson* define Data Integrity as follows:

Definition 69 : Data Integrity by *Clark and Wilson*

> *"**Data Integrity**: No user of the system, even if authorized, may be permitted to modify data items in such a way that assets or accounting records of the company are lost or damaged.*
> *[...]*
> *The concept of the well-formed transaction is that a user should not manipulate data arbitrarily, but only in constrained ways that preserve or ensure the* ***integrity*** *of the data." [125]*

Clark and Wilson **explicitly allow authorized subsequent modifications.** *Clark and Wilson* acknowledge the existence of modifications which do not harm integrity. They clearly state that there are "well-formed transactions" [125], these would allow to "manipulate data [...] only in constrained ways" [125] such that the modification does not "damage the asset" [125]. Damage of an asset is the security threat that *Clark and Wilson*'s integrity notion protects against.

Clark and Wilson describe Integrity as being protected by prohibitive mechanisms, i.e. integrity protection is seen as a preventative mechanisms. Thus, no explicit statements regarding detection of subsequent authorized modifications are made.

5.1.4. Integrity by ISO 2700x series

The ISO 2700x series is on Information Security Management Systems (ISMS). ISO 27000, in the fourth and latest edition from 2016, contains the overview and the vocabulary for the series. In the version from 2005 the term 'integrity' was listed as part of the term 'information security' defined by ISO 27001 [246] alongside terms like 'availability' and 'confidentiality'.

In respect to integrity the following wording can be found throughout the version of ISO 27000 from 2005 till 2016:

[247] The American English spelling from original work is retained for terms and quotes.

[248] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[249] "External consistency: The correspondence between the data object and the real world object it refers to" [125]

> “The term information security is generally based on information being considered as an asset which has a value requiring appropriate protection, for example, against the loss of availability, confidentiality and integrity. Enabling accurate and complete information to be available in a timely manner to those with an authorized need is a catalyst for business efficiency.” [246, 249, 254, 260][250]

In the edition of ISO 27000 from 2009 there was the following definition of integrity:

Definition 70 : Integrity from ISO 27000:2009

> *“**integrity***
> *property of protecting the accuracy and completeness of assets” [249]*

This was followed up by the second edition in 2012 and the third edition was published in January 2014. In the version of ISO 27000 from 2014 and also in the very latest version from 2016 the definition of integrity is as follows:

Definition 71 : Integrity from ISO 27000:2016

> *“**integrity***
> *property of accuracy and completeness” [254, 260]*

The definition of integrity in the ISO 2700x series got more and more reduced, and this notion of integrity makes a statement whether or not the data’s quality is suitable for the application, e.g., “accurate” [246] or “property of accuracy” [260]. This is related to the notion of veracity given by *Gollmann* in [215] discussed in Sec. 5.1.8.

ISO 2700x series implicitly tolerates authorized modifications. The definitions from the ISO 2700x series do not explicitly forbid modifications as long as they keep the data’s “accuracy and completeness” [260]. Hence, the ISO 2700x series implicitly allows subsequent modifications.

Further, the ISO 2700x series makes no explicit statements regarding the detection of subsequent authorized modifications.

5.1.5. Data integrity from ISO 7498-2 and ITU X.800

ISO 7498-2 [239] and the ITU X.800 [263] are technically aligned and offer identical definitions regarding data integrity.

Definition 72 : Data integrity by ISO 7498-2 and by ITU X.800

> *“The property that data has not been altered or destroyed in an unauthorized manner.” [239, p. 3] [263]*

The ITU X.800 defines the attack against integrity as follows:

Definition 73 : Modification of messages by ITU X.800

> *“Modification of a message occurs when the content of a data transmission is altered without detection and results in an unauthorized effect, as when, for example, a message "Allow ’John Smith’ to read confidential file ’Accounts’" is changed to "Allow ’Fred Brown’ to read confidential file ’Accounts’." ” [263]*

[250] During the study of literature related to the term of integrity, the following quoted text appeared several times: “[...] Integrity. Assurance that the information is authentic and complete. Ensuring that information can be relied upon to be sufficiently accurate for its purpose. [...] The integrity of data is not only whether the data is ’correct’, but whether it can be trusted and relied upon; [...]” [334]. The above can be found in the glossary of a recent EU project’s deliverable [334] where it is sourced as “[ISO27001]” [334]. However this is incorrect, as this thesis found that this text as quoted is neither contained in the 2005 version [246] nor in the 2013 version [252] of ISO 27001. This wrong attribution also appeared in a recent book from 2017 [404, p. 310] where the cite is sourced as “ISO 2700”, which is a different and deceased standard related to road vehicles from 1974. The quote itself can also be found online `http://security.globalpractitioner.org/introduction/infosec_2.htm` [last accessed: Jan. 2017] or at `http://i40.iosb.fraunhofer.de/FA7.21Begriffe-EigenschaftenIndustrie4.0-Komponente` [last accessed: Jan. 2017]. It also appears to be sourced as “part of the RUsecure Security Policy Suite” as mentioned on this web page `http://www.yourwindow.to/information-security/gl_confidentialityintegrityandavailabili.htm` [last accessed: Jan. 2017].

Further, ISO 7498-2 and ITU X.800 define digital signatures as one tool "to prove the [...] integrity of the data unit" [239] using cryptography. The latter is defined as the "discipline which embodies principles, means, and methods for the transformation of data in order to [...] **prevent its undetected modification** [...]"[251] [239, 263].

Further, the ITU X.800 and the ISO 7498-2 describe five services that are different in their scope (i.e., message vs. selective fields) and different in whether or not they attempt to recover from errors, i.e., provide error correction features. The first service's definition by [263] is as follows:

Definition 74 : Data integrity services from ITU X.800

> *"5.2.4 Data integrity*
> *These services counter active threats and may take one of the forms described below.*
> *Note – On a connection, the use of the peer entity authentication service at the start of the connection and the data integrity service during the life of the connection can jointly provide for the corroboration of the source of all data units transfered on the connection, the integrity of those data units, and may additionally provide for the detection of duplication of data units, e.g., by the use of sequence numbers.*
>
> *5.2.4.1 Connection integrity with recovery*
> *This service provides for the integrity of all (N)-user-data on an (N)-connection and **detects any modification**, insertion, deletion or replay of any data within an entire SDU sequence (with recovery attempted).*
>
> *[...]"[251] [252] [263]*

ITU X.800 implicitly tolerates authorized modifications. ITU X.800 requires detecting any modification. The thesis sees ITU X.800's understanding of the notion of a modification as follows: A modification is some event that is not being detected and that must always result in an "unauthorized effect" [263]. In contrast to the description of the services, the integrity definition of the ITU X.800's explicitly uses the word "unauthorized" [263] in the integrity definition.

The described integrity services achieve detecting "any modification, insertion, deletion or replay of any data" [263] or "whether the selected fields have been modified, inserted, deleted or replayed" [263]. Here, the ITU X.800's "any" in "detects any modification" [263] is very explicit. It explicitly requires the detection of subsequent authorized — and unauthorized — modifications. However, it does not prohibit a differentiation between them during detection.

As the contents of ITU X.800 are technically aligned with the contents of ISO 7498-2 the analysis yields no different results.

ISO 7498-2 implicitly tolerates authorized modifications. ISO 7498-2 requires detecting any modification. The definition found in ISO 7498-2 includes the notion of "unauthorized manner" [239]. Thus, ISO 7498-2 implicitly tolerates authorized modifications, as they do not alter data in an "unauthorized manner" [239].

ISO 7498-2 further defines five different data integrity services all of them requiring that the service "[...] detects any modification, insertion, deletion or replay of data [...]" [239, p. 5]. Hence, ISO 7498-2 explicitly requires that data integrity services enable the detection of "any modification" [239, p. 5], which includes subsequent authorized modifications. ISO 7498-2 still allows differentiating in the detection between authorized and unauthorized modifications.

5.1.6. Data integrity in RFC 4949

The IETF's Internet Security Glossary (RFC 4949) gives the following description:

Definition 75 : Data integrity from RFC 4949

> *"data integrity*

[251] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.
[252] The American English spelling from original work is retained for terms and quotes.

> 1. *(I) The property that data has not been changed, destroyed, or lost* ***in an unauthorized*** *or accidental* ***manner****. (See: data integrity service. Compare: correctness integrity, source integrity.)*
>
> 2. *(O) "The property that information has not been modified or destroyed in an unauthorized manner." [I7498-2]*
>
>   *Usage: Deals with (a) constancy of and confidence in data values, and not with either (b) information that the values represent (see: correctness integrity) or (c) the trustworthiness of the source of the values (see: source integrity)."*[253] *[438]*

The second description in RFC 4949 is marked with "(O)" which indicates that it is not recommended to use. The difference to the recommended notion is subtle, most notably it includes "loss" and "accidental" in the protection scope. Noteworthy, the RFC clearly states that their notion of data integrity does not allow deriving statements about the origin's trustworthiness nor the values' correctness. The same has as well been differentiated from integrity by the notion of veracity given by *Gollmann* (discussed in Sec. 5.1.8).

RFC 4949 further defines data integrity service as follows:

> "A security service that protects against **unauthorized changes** to data, including both intentional change or destruction and accidental change or loss, by **ensuring that changes to data are detectable**."[253] [438]

This thesis interprets the above as an explicit statement that in order to offer the protection it is required that all "changes to data are detectable" [438], including authorized subsequent modifications.

Further RFC 4949 explicitly states in its tutorial text that data integrity services "can only detect a change" [438] and that "changes cannot be prevented" [438]. Noteworthy is the following, limiting statement of RFC 4949: "The ability of this service to detect changes is **limited by the technology of the mechanisms used** to implement the service."[253] [438]. Hence, RFC 4949 acknowledges that it is the mechanism's actual algorithmic description and the implementation details that will in the end define what modifications are detected.

EXAMPLE

RFC 4949 gives the following example: "[...] [I]f the mechanism were a one-bit parity check across each entire [service data unit] SDU, then changes to an odd number of bits in an SDU would be detected, but changes to an even number of bits would not." [438].

RFC 4949 implicitly tolerates authorized modifications. RFC 4949 explicitly requires all subsequent modifications to be detectable. The definition found in RFC 4949 includes the notion of "unauthorized or accidental manner" [438]. Assuming that authorized modifications to be not of that 'manner', RFC 4949 implicitly tolerates authorized modifications not invalidating the integrity.

RFC 4949 clearly aims for the detection of any subsequent modification as "ensuring that changes to data are detectable" [438]. This thesis interprets this as an explicit statement that all subsequent — including authorized — "changes to data are detectable" [438].

Noteworthy, RFC 4949 explicitly acknowledges that the chosen technical mechanism defines[a] what is detected. Alongside, the chosen technical mechanism defines what constitutes an unauthorized modification and whether or not authorized modifications are detectable.

[a] RFC 4949 formulates it as being "limited by" [438].

5.1.7. Integrity of network messages by *Stallings*

Stallings defines Data Integrity with respect to a message stream or single message or parts thereof in two forms: connection-oriented and connectionless integrity services [452]. This is, like the ITU X.800 definition, in the view of a layered network stack.

Definition 76 : Integrity of network messages by *Stallings*

[253] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

> "A connection-oriented integrity service*, one that deals with a stream of messages, assures that messages are received as sent, with no duplications, insertion, modification, reordering, or replays.*
> *[...]*
> *Thus, the* connection-oriented integrity service *addresses both message stream* ***modification*** *and denial of service.*
> *[...] a* connectionless integrity service*, one that deals with individual messages only without regard to any larger context, generally provides protection against message* ***modification*** *only."*[254] *[452]*

Stallings definition targets actual physical and logical network layers that are involved in a transmission of a message. Network transmission is seen as a layered approach, and modifications are only those which are not reverted at the receiver's layered stack.

***Stallings*'s Definition 76 explicitly forbids authorized modifications, even if detected.** *Stallings*'s definition makes no distinction when describing the modifications that are not allowed, no reference to an unauthorized or malicious modification. *Stallings*'s integrity definition is very explicit that integrity means that "messages are received as sent" [452].

Stallings sees an integrity service as means to allow detecting the data's availability, which is threatened by loss or destruction. Thus integrity services are mainly concerned "with detection rather than prevention" [452]. This means that all subsequent modifications, that are not reverted, must be detected, otherwise integrity would be violated as messages must be received as sent.

5.1.8. Data integrity and veracity by *Gollmann*

Gollmann separates protection mechanisms for the security protection of assets into three classes: prevention, detection and reaction [214]. For integrity this thesis is concerned with the detection of integrity. *Gollmann* starts the definition of his integrity notion with the following statement:

Definition 77 : Data integrity in general by *Gollmann*

> *"In general, integrity is about making sure that everything* ***is as it is supposed to be****."*[254] *[214, p. 35, 3^{rd} ed.], [213, p. 21, 2^{nd} ed.]*

Gollmann defines the result of integrity check functions in [213, 2^{nd} ed.] and [214, 3^{rd} ed.] as follows:

Definition 78 : Data integrity check functions by *Gollmann*

> *"[...]* ***Data Integrity****: integrity check functions provide the means to* ***detect*** *whether a document has been changed;"*[254] *[213, p. 187, 2^{nd} ed.], [214, p. 253, 3^{rd} ed.].*

Gollmann sees data integrity as a prerequisite[255] and somewhat coupled with data origin authentication, but also clearly states that "[...] a separate notion of Data Integrity makes sense in other applications, e.g. for file protection in anti-virus software." [213, 2^{nd} ed.], [214, p. 253, 3^{rd} ed.]. See Sec. 5.2.6 for a brief discussion of the authenticity and non-repudiation definitions from *Gollmann*.

Gollmann elaborates that integrity[256] is not verifying that "[t]he data are already false when passed to the infrastructure [...] " [215]. The author then calls this "veracity" [215]. Veracity is "[t]he property that [a statement][257] truthfully reflects the aspect it makes a statement about" [215]. *Gollmann* notes that "[...] in the field of databases veracity corresponds to external consistency [...]" [215].

[254] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.
[255] "Integrity is often a prerequisite for other security properties." [213, p. 21, 2^{nd} ed.], [214, p. 36, 3^{rd} ed.].
[256] Additionally to integrity *Gollmann* also mentions authentication and non-repudiation; which is discussed briefly in Sec. 5.2.6.
[257] In the original text this is called "an assertion" [215].

***Gollmann*'s Definition 77 implicitly tolerates authorized modifications. Detection of any subsequent modification is explicitly required.** When mentioning the scope of detection that integrity check functions are considering, *Gollmann*'s book does not make any descriptive exclusion or inclusion and specifies the need to "[...] detect whether a document has been changed [...]" [213, p. 187, 2nd ed.]. Regarding, scope and goals of integrity, *Gollmann* notes further that integrity is not veracity and thus integrity does not prohibit that "[t]he data are already false when passed to the infrastructure [...]" [215] for integrity protection.

This thesis considers that modifications that have been authorized are covered by *Gollmann*'s general notion of "supposed to be" [214]. In more detail *Gollmann* states in the third edition of his book that "[...] in communications security, integrity refers to the detection [...] of modifications [...]" [214] which '[...] includes both intentional manipulations and random transmission errors [...]" [214]. *Gollmann* continues that one "[...] could view intentional modification as a special case of unauthorized modification when nobody is authorized to modify." [214, p. 36, 3rd ed.]. Note, this is seen from the perspective that the intentional modification is indeed an unwanted manipulation when "[...] nobody is authorized to modify." [214]. It would additionally require to discuss if an intentional modification is welcome and would thus be authorized. This additional authorization is however not discussed in the Definitions 77 and 78 by *Gollmann*. The above quotes on communication security terminology are discussed by *Gollmann* in the light of "[...] add[ing] to the confusion [...]" [214, p. 36, 3rd ed.] around the term integrity.[a] *Gollmann* continues to acknowledge that to identify what is unauthorized, it is required that "[...] an authorization structure has an impact on the nature of the problem that has to be solved, and on the respective security mechanisms." [214, p. 36, 3rd ed.].[b] This added weight to the need to identify integrity in general as being policy controlled to adapt to applications as noted in Sec. 5.5.1.

Overall, Definition 77 on integrity in general, as well as Definition 78 on data integrity are seen as to implicitly tolerate authorized modifications.

Regarding detection of authorized subsequent modifications, this thesis assumes that *Gollmann*'s notion of "changed" contains all modifications and hence authorized modifications are subsumed. In both definitions (Definition 77 and Definition 78), *Gollmann* explicitly requires the **detection of any modification**. Thus, detection of any subsequent modification is explicitly required by *Gollmann*.

[a] *Gollmann* initially starts his book chapter on integrity stating that "[i]t is not easy to give a concise definition of integrity." [214, p. 35, 3rd ed.].

[b] Note that *Gollmann* sets this in a more negative light by stating the above as follows: "You could view intentional modification as a special case of unauthorized modification when nobody is authorized to modify. However, there is not much to gain from taking such a position because the presence, or absence, of an authorization structure has an impact on the nature of the problem that has to be solved, and on the respective security mechanisms." [214, p. 36, 3rd ed.]. This thesis discusses in detail in Sec. 5.5.1 the need to have such a policy in order to differentiate authorized from un-authorized and therefore to-be-detected modifications. This thesis acknowledges the problem that any authorization of subsequent modifications increases the complexity of the integrity policy, i.e., it is easier to have the list of authorized modifications empty and detect any subsequent modification due to it being unauthorized.

5.1.9. Data integrity by *Bishop*

In his book *Bishop* stated that "[Integrity] is usually phrased in terms of preventing improper or unauthorized change." [51, p. 5]. He separates "Integrity mechanisms [...] into two classes: prevention mechanisms and detection mechanisms." [51, p. 5]. This thesis's scope is the detection, which *Bishop* describes to "[...] report the actual cause of the integrity violation (a specific part of the file was altered), or [...] report that the file is now corrupt [...]" [51, p. 5].

Bishop states the threat against which integrity services protect is as follows:

Definition 79 : Data integrity by *Bishop*

> *"Modification or alteration, an* ***unauthorized change*** *of information, cover three classes of threats. [...] Integrity services counter this threat."*[258] *[51, p. 7]*

Regarding the definition of what is authorized and what is not, *Bishop* further states that:

[258] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

Definition 80 : Integrity policy by *Bishop*

> *"With respect to integrity, a security policy identifies* ***authorized ways in which information may be altered*** *and entities authorized to alter it. [...] Those parts of the security policy that describe the conditions and manner in which data can be altered are called the integrity policy."[259] [51, p. 97]*

Bishop**'s definitions (Definition 79 and Definition 80) explicitly allow specifying authorized subsequent modifications using a security policy.** The explanations by *Bishop* for integrity do not allow an "unauthorized change" [51] of integrity-protected data. This is ascribed to *Bishop*'s idea of integrity to also be preventative. In *Bishop*'s definition regarding security policies for integrity the authorized subsequent modification gets explicitly acknowledged: Integrity could allow for "authorized ways in which information may be altered" [51] if the integrity policy allows for that.

Regarding detection, *Bishop* separates mechanisms that detect integrity breaches and mechanisms that prevent violations of integrity. "Detection mechanisms do not try to prevent violations of integrity; they simply report that the data' integrity is no longer trustworthy."[51, p. 5]. The definition does not require to detect subsequent modifications that comply to integrity policies and thus will not negatively affect the data's integrity. This thesis concludes that *Bishop* is not explicitly requiring any subsequent modification to be detectable.

5.1.10. Integrity framework from ISO 10181-6

According to the classification of integrity services from ISO 10181-6 [340] integrity protection mechanisms can allow the "detection of integrity compromise" [340] or the "prevention" [340] of integrity breaches. This thesis' scope is on detection mechanisms (see Sec. 1.4.1). Within the integrity framework of ISO 10181-6 this equals to protection against all of the following three out of five violations:

"a) unauthorized data modification;

[...]

c) unauthorized data deletion;

d) unauthorized data insertion; [...]" [340].

All above listed violations are explicitly termed as being "unauthorized" [340].

ISO 10181-6 sees integrity protection as several operations; all these options could be seen as several algorithms within an cryptographic scheme. Among them are "shield" [340], which "[...] applies integrity protection to data [...]" [340] and "validate", which "[...] checks integrity-protected data for modifications [...]" [340]. ISO 10181-6 further offers a classification of integrity mechanisms in four classes:

Class 1 "Integrity provision through cryptography" [340]

Class 2 "Integrity provision through context" [340]

Class 3 "Integrity provision through detection and acknowledgement" [340]

Class 4 "Integrity provision through prevention" [340]

The analysis of the descriptions and limitations of the classes yielded that only Class 1, and in particular only the "Integrity provision through Digital Signatures" [340] will offer a standalone protection of integrity verifiable by third-parties. The mechanisms for Class 1 mentioned are: symmetric key based "sealing" [340], encipherment of "Redundant Data (e.g. natural language)" [340], and "Digital Signatures [...] computed using a private key and an asymmetric cryptographic algorithm" [340].

Class 2 requires data to be replicated and stored or transmitted in pre-agreed contexts. ISO 10181-6 states limitations, like the assumption that adversaries cannot compromise multiple replicated storage locations such that the data can no longer be re-constructed or recommend that the mechanisms "must [...] be combined with other integrity mechanisms" [340].

Class 3 also has severe limitations as ISO 10181-6 states that they are "not, as a rule, suitable for data storage" [340] or "assume that data modifications can be detected through other means" [340].

[259] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

Class 4 mechanisms provide integrity by “preventing physical access to data storage or transmission media and through access control” [340]. In ISO 10181-6 the Class 4 mechanisms are preventative.

ISO 10181-6 implicitly tolerates authorized modifications. The definition given in ISO 10181-6 requires generating evidence enabling an operation that “checks integrity-protected data for modifications” [340]. ISO 10181-6 states that digital signatures are mechanisms that allow adequate stand alone protection (i.e., Class 1). ISO 10181-6 also list the advantage that “Digital Signatures permit the set of entities which can validate the data to be arbitrary in size and composition.” [340]. Further, to generate evidence towards untrusted third-parties, ISO 10181-6 rules out symmetric keys as evidence mechanisms, because with a symmetric key “the set of entities capable of sealing the data and the set of entities capable of validating the data are [...] coincident” [340].

ISO 10181-6 makes no explicit statements regarding the detection of authorized subsequent modifications.

5.2. Authorized modifications in existing technical definitions regarding (data) origin authentication and non-reputation

This section analyses what technical definitions of non-repudiation and authentication of origin state with respect to authenticity for information that has undergone authorized subsequent modifications. As previously shown, it is important for the legal recognition of the signed content to technically identify and proof which entity had issued the content of a message (see Chapter 4). The definitions from *ISO 7498-2*, *ISO 13888-1*, *ISO 10181-4*, *RFC 4949*, *ITU X.800* and *Gollmann* are analysed.

5.2.1. Data origin authentication from ISO 7498-2 and ISO 13888-1

ISO 7498-2 [239] has defined the notion of data origin authentication. The following definition is a quote from ISO 13888-1 [243] or RFC 4949 [438].

Definition 81 : Data origin authentication from ISO 7498-2

> *“corroboration that the source of data received is as claimed” [239]*

ISO 7498-2 (and ISO 13888-1) require the technical mechanisms to provide a way to verify the claim of the data’s origin. To change the claimed origin from the Signer to the Sanitizer requires a subsequent authorized modification to be detectable. Once a subsequent modification happened, a claim of the Signer being accountable for the modifications would no longer be corroborated by the mechanism. If the claim changes to the Sanitizer after a subsequent modification, then the Sanitizer must become accountable.

ISO 7498-2 (and ISO 13888-1) implicitly tolerate detectable and Sanitizer-accountability authorized modifications as a mechanism for Data Origin Authentication; it is required to allow a verification of the change of claimed origin. Hence, after data has undergone any, even authorized, modifications, the Signer will no longer be corroborated as a claimed source and data Origin Authentication as defined by ISO 7498-2 (and ISO 13888-1) would not be given. However, if through detection the Verifier can corroborate the Sanitizer as “source of data received”, then data origin authentication, as defined by ISO 7498-2 and ISO 13888-1, can still be established for subsequently modified data.

ISO 7498-2 (and ISO 13888-1) Data Origin Authentication mechanism do not explicitly require the accountability of the Sanitizer for any authorized subsequent modification done by the Sanitizer. ISO 7498-2 and ISO 13888-1 do not explicitly state that the new source needs to become accountable. The reason for this is that the claim must not be updated by the mechanisms for data origin authentication. Namely, the mechanism is only required to enable that after any subsequent modification the originally claimed source, e.g. the Signer, cannot be framed as being the source for data received which has subsequently been modified.

5.2.2. Data origin authentication from ITU X.800

The ITU X.800 offers a definition of data origin authentication explicitly geared towards layered communication stacks.

Definition 82 : Data origin authentication from ITU X.800

> *"This service, when provided by the (N)-layer, provides corroboration to an (N + 1)-entity that the source of the data is the claimed peer (N + 1)-entity. The data origin authentication service provides the* ***corroboration of the source of a data unit****. The service* ***does not provide protection against*** *duplication or* ***modification*** *of data units."*[260] *[263]*

Above, ITU X.800 notion of origin authentication does not protect against modification of data, hence it allows knowing the origin of the message, but not if the received message was the one that was sent by the originator. As usually both is needed, X.800 suggests to combine entity authentication explicitly with an integrity protection service.

Definition 83 : Suggested combination of integrity and entity authentication from ITU X.800

> *"Note – On a connection, the use of the peer entity authentication service at the start of the connection and the data integrity service during the life of the connection can jointly provide for the corroboration of the source of all data units transfered on the connection, the integrity of those data units, and may additionally provide for the detection of duplication of data units, e.g. by the use of sequence numbers."*[261] *[263]*

ITU X.800 explicitly allows modifications not affecting Data Origin Authentication. ITU X.800 requires the technical mechanism to provide a method to verify claims of the data's origin, but explicitly "[...] does not provide protection against [...] modification [...]" [263]. Sec. 5.1.5 found ITU X.800 integrity to implicitly tolerate that authorized modifications do not breach data integrity. Hence, to provide data origin authentication following ITU X.800, the mechanism does not need to detect subsequent authorized modifications nor does the mechanism need to make the Sanitizer accountable for the modifications. Thus, if the Verifier can corroborate an entity, be it the Sanitizer or the Signer, as "source of the data unit", then data origin authentication, as defined by ITU X.800, is provided even if data has undergone any – including authorized — modification.

5.2.3. Non-repudiation with proof of origin from ITU X.800

According to the ITU X.800 [263] non-repudiation may take one or both of two forms: Non-reputation with proof of origin or non-reputation with proof of delivery. This thesis is only concerned with non-reputation with proof of origin.

Definition 84 : Non-repudiation with proof of origin by ITU X.800

> *"The recipient of data is provided with proof of the origin of data. This will protect against any attempt by the sender to falsely deny sending the data or its contents." [263]*

Following the above definition of non-repudiation with proof of origin by ITU X.800, a mechanism would require that the Verifier can corroborate the Signer as "[...] the origin of data [...]" [263] for a message that was not modified. As such the Verifier is provided the assurance that the data has not undergone any — unauthorized and authorized — subsequent modifications. To fulfil non-repudiation with proof of origin when the data has been subject to authorized modifications the mechanism must attribute the data to originate from the Sanitizer. Thus, for an authorized modified message the Verifier must corroborate the Sanitizer as "the origin of data", because the Signer would be able to refute being the source as the data was subject to subsequent modifications. then non-repudiation with proof of origin, as defined by ITU X.800, is provided.

[260] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.
[261] The American English spelling from original work is retained for terms and quotes.

ITU X.800 explicitly allows repudiation of origin once the "contents" [263] of the message has indeed been modified. ITU X.800 non-repudiation with proof of origin prohibits the sender to refute having sent "[...] the data or its contents [...]" [263]. It explicitly refers to the "contents" [263], thus would require to allow refuting being the origin, i.e. having applied integrity protection, for a message that has been subsequently modified in any way. Hence, the Signer must not be able to repudiate unmodified messages, the Sanitizer must not be able to repudiate authorized modified messages, and both must be able to repudiate unauthorized modified messages.

5.2.4. Non-repudiation with proof of origin from RFC 4949

The resulting functionality of non-repudiation is described in RFC 4949's tutorial for the term "non-repudiation with proof of origin" [438]. RFC 4949 defines the service provided by a mechanism that achieves non-repudiation of origin as follows:

Definition 85 : Non-repudiation with proof of origin from RFC 4949

> *" A security service that provides the recipient of data with evidence that proves the origin of the data, and thus protects the recipient against an attempt by the originator to falsely deny sending the data." [438]*

In the RFC 4949 it is accompanied by the following tutorial:

> "Tutorial: This service is a strong version of data origin authentication service. This service can not only verify the identity of a system entity that is the original source of received data; it can also provide proof of that identity to a third party." [438]

The referenced data origin authentication service is defined as follows:

Definition 86 : Data origin authentication service from RFC 4949

> *"A security service that verifies the identity of a system entity that is claimed to be the original source of received data." [438]*

Furthermore, RFC 4949 describes a relationship between data integrity and data origin authentication as follows:

> "Although data integrity service is defined separately from data origin authentication service [...], it is closely related to them. Authentication services depend, by definition, on companion data integrity services. Data origin authentication service provides verification that the identity of the original source of a received data unit is as claimed; there can be no such verification if the data unit has been altered." [438]

Hence, RFC 4949 sees data integrity as a pre-requisite for origin authentication[262].

In the absence of authorized or unauthorized modifications RFC 4949's "original source of received data" [438] is interpreted as being able to identify the Signer as the "original source" [438]. However, from the text of non-repudiation it remains unclear what happens in case of an authorized subsequent modification. This would result in a difference in the "received data" but depending on the accountability and detectability the Signer would be able to deny being the "original source" of that modified data. With a very strong interpretation of the adjective 'original' in the tutorial's phrase "original source" [438], the interpretation would be: RFC 4949 tolerates the verification to point to the Signer even after a modification. However, the latter interpretation would contradict the thesis' interpretation of RFC 4949's data integrity definition: While it has been analysed to tolerate subsequent modifications it demands them being detectable (see Sec. 5.1.6). With even authorized subsequent modifications being detectable, the evidence that "received data", which was detected as subsequently modified, was from the Signer could not be uphold towards a third party that also detected the subsequent modification. RFC 4949's non-repudiation definition would work if after a subsequent modification the Signer is no longer the appointed source, but the Sanitizer is.

[262] Same as *Gollmann* in [214, 3^{rd} ed.].

RFC 4949 definition of non-repudiation — in the light of the interpretation of RFC 4949's data integrity definition — allows rebuttal of the evidence of being the source for modified data. RFC 4949 definition of non-repudiation requires to appoint the Sanitizer as origin after an authorized modification. RFC 4949's data integrity definition was interpreted as demanding all subsequent modifications, including authorized modifications, as being detectable. RFC 4949's "original source of received data" [438] hence would no longer be evidently the Signer. Hence, in case of authorized modifications to the data before it was received a non-repudiation mechanisms following RFC 4949 would require to identify the Sanitizer as the new "[...] source" [438].

5.2.5. Non-repudiation of origin by ISO 10181-4 and ISO 13888-1

The general goal of services for non-repudiation is given in ISO 10181-4 as follows:

Definition 87 : Goal of non-repudiation service by ISO 10181-4

> *"The goal of the Non-repudiation service is to collect, maintain, make available and validate irrefutable evidence concerning a claimed event or action in order to resolve disputes about the occurrence or non-occurrence of the event or action." [241]*

ISO 13888-1 states the goal only slightly differently as follows:

> 'The goal of a non-repudiation service is to generate, collect, maintain, make available and verify evidence concerning a claimed event or action in order to resolve disputes about the occurrence or non occurrence of the event or action." [243, p. V]

The above goal description was selected as the thesis general definition of the term 'non-repudiation service':

Definition 88 : Non-Repudiation Service following ISO 13888-1

> *"[...] [A] non-repudiation service [...] generate[s] [...] and verif[ies] evidence concerning a claimed event or action in order to resolve disputes about the occurrence or non occurrence of the event or action." [243]*

The establishment of evidence that a message originated from a source which cannot be repudiated is technically termed "non-repudiation of origin". In ISO 13888 which is the dedicated ISO standard for security techniques that achieve this, it is defined as follows:

Definition 89 : non-repudiation of origin by ISO 13888-1

> *"service intended to protect against the originator's false denial of having created the content of a message and of having sent a message" [243]*

It defines a non-repudiation services as follows:

Definition 90 : Non-repudiation services by ISO 13888-1

> *"Non-repudiation involves the generation of evidence that can be used to prove that an event or action has taken place. Evidence is generated in the form of verifiable data describing the actions or events." [243]*

Further, ISO 13888-1 states that the link to the entity is done via the cryptographic key material when using digital signatures.

Definition 91 : Evidence verification mechanism from ISO 13888-1

> *"The provision of the public verification key to the verifier depends on the type of signature scheme applied to generate the digital signature.*
>
> - *Certificate-based signatures are verified using the public key of the signer available in the public key certificate issued by the certification authority (CA).*
>
> *[...]" [243]*

In the mechanism for non-repudiation of origin as defined by ISO 13888-1 it is stated that such a mechanism allows the Verifier to corroborate the Signer as "[...] having created **the** content [...]"[263] [243]. ISO 13888-1's wording leads this thesis also to interpret that this mechanism must provide assurance that the data has not undergone any — neither unauthorized nor authorized — modification if it appoints the Signer as the "originator" [243]. To fulfil non-repudiation of origin, as defined by ISO 13888-1, for data that has been subject to authorized modifications, the mechanism for the "Evidence verification phase" [243] of non-repudiation must attribute the data to originate from the Sanitizer. Thus, this thesis sees that an ISO 13888-1–compliant Verifier must corroborate the Sanitizer as the "originator" [243] of data whenever the data was subject to subsequent authorized modifications. Both, Sanitizer and Signer must be able to refute if data has undergone unauthorized subsequent modifications. Then non-repudiation of origin, as defined by ISO 13888-1, is provided.

ISO 13888-1 explicitly allows repudiation of origin by the Signer once the "content of a message" [243] has indeed been subjected to subsequent modification. ISO 13888-1 defines a notion of non-repudiation of origin that prohibits the sender from refuting "[...] having created the content of a message and of having sent a message [...]" [243] if the content of a message is unmodified. It explicitly refers to the "content" [243] as it is received at the Verifier thus would require to allow refuting being the "originator" [243], i.e., having applied non-repudiation of origin protection, for a message that has been modified in any way.

In summary: The Signer must not be able to repudiate unmodified messages, but can repudiate modified messages including authorized modified ones. The Sanitizer must not be able to repudiate authorized modified messages, but can repudiate unmodified and unauthorized modified ones.

5.2.6. Data origin authentication and non-repudiation by *Gollmann*

Gollmann noted that "[a]uthentication and non-repudiation potentially take us away from a pure IT-centric view of security." [215] Note that this thesis's analysis of the legal notions of IT security fully agrees with this and Analysis Result 5 recorded the legal importance of accountability. *Gollmann* sees data integrity as a pre-requisite for origin authentication[264], and states "[y]ou cannot claim to have verified the source of a message that has been changed in transit." [213, 2nd ed., p. 187] and that "[a] message that has been modified in transit no longer comes from its original source." [214, 2nd ed., p. 253]. *Gollmann* gives the following definition of origin authentication which couples the provision of the former with the verification of integrity:

Definition 92 : Data origin authentication by *Gollmann*

> *"data origin authentication: [...] provide the means to verify the source and integrity of a message." [214]*

For certain cases *Gollmann* acknowledges that "a separate notion of Data Integrity makes sense in other applications" [214].

Noteworthy, in [215] *Gollmann* introduces the notion of veracity of data. In short, veracity is the property that a statement[265] contained in electronic data "[...] truthfully reflects the aspect it makes a statement about [...]" [215]. *Gollmann* states that "[...] authentication and non-repudiation verify the claimed origin of [statements] [...]" [215] and that those statements[265] "[...] may be true or false [...]" [215].

***Gollmann* couples the protection of origin authentication with integrity protection. Both notions are said to hold regardless of the truthfulness of the information contained (separate notion of veracity).** The notion of data origin authentication following *Gollmann* requires allowing the verification of the message's source and at the same time also its integrity.

[263] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[264] Alike RFC 4949.

[265] Originally *Gollmann* called them assertions: "Assertions are statements about an aspect relevant in a given application domain." [215].

The notion of veracity of data is important as it explicitly allows data origin authentication and integrity to hold for data that is false on purpose or false by accident in its electronic representation *at the time* the protection mechanism was applied. *Gollmann* states that the protection of authentication of origin and integrity does not cover statements[a] "that are false on purpose but includes also [statements][265] that are false by accident" [215]. Noteworthy, *Gollmann*'s notion covers both, accidentally and maliciously false statements, because "[t]his distinction does not matter for the application; the application would react to a false [statement][265] in the same way in both cases." [215]. The former also contributes to Analysis Result 8.

[a] Originally *Gollmann* called them assertions: "Assertions are statements about an aspect relevant in a given application domain." [215].

5.3. Authorized modifications in existing technical definitions of classic digital signature schemes (CDSS)

This section analyses what statements technical definitions of digital signatures make regarding the authenticity level of information that has undergone authorized subsequent modifications. It will consider technical definitions from cryptographic literature as well as from those standards or recommendations that legal texts have referenced directly or indirectly. The US technical standard NIST FIPS 168-4 is directly mentioned in FISMA, as discussed in Sec. 4.3.4.2. The same standard is also mentioned in the German catalogue of algorithms, which is issued at least yearly due to German legal texts SigVO. The catalogue of algorithms implements a requirement of SigVO that technically details the SigG, which in turn implements Directive 1999/93/EC. This chain from legal texts to technical standards is also depicted in Fig. 26. To be specific, in this section the definitions from ISO 7498-2, ITU X.800, NIST FIPS 168-4, PKCS#1_v2.2 and the cryptographic definition by *Goldwasser*, *Micali*, and *Rivest* are analysed.

The discussion of the actual parameters or the secure implementation of algorithms is out of scope of this thesis. This thesis will not go into details on the key lengths or implementation details that are mentioned in various standards and papers for certain algorithms. The interested reader is referred to the work done by *Loebenberger and Nüsken* [319] for a discussion of the most known standards for RSA integers and their differences, e.g.,PKCS#1, IEEE 1363-2000, FIPS 186-3, ANSI X9.44, or ISO 18033-2.

5.3.1. Germany: Catalogue of algorithms ('Algorithmenkatalog') by BNetzA

In Germany the 'Bundesnetzagentur'[266] (BNetzA) has been named in the former SigVO[267] as the organisation that every year publishes the authoritative list of algorithms and their parameters that are deemed suitable to generate qualified electronic signatures[268]. In the now passed VDGfollowing Regulation 910/2014 the BNetzA is still listed[269] Being in this *catalogue of algorithms*[270] means that the algorithm is legally recognised. In Germany the report is published periodically on a yearly basis, e.g., [84, 85]. [271]

[266] 'Bundesnetzagentur' translates to 'Federal Network Agency for Electricity, Gas, Telecommunications, Post and Railway' in German.

[267] "Anlage 1 Abschnitt 1 Nr. 2 Signaturverordnung (SigV)" [84].

[268] In German it reads: "Die zuständige Behörde veröffentlicht im Bundesanzeiger eine Übersicht über die Algorithmen und zugehörigen Parameter, die zur Erzeugung von Signaturschlüsseln, zum Hashen zu signierender Daten oder zur Erzeugung und Prüfung qualifizierter elektronischer Signaturen als geeignet anzusehen sind [...]" [84].

[269] The BNetzA has still the duty to maintain the catalogue; in German the website on the duties states "Ferner veröffentlicht sie jährlich einen Katalog von Algorithmen, die für die Nutzung im Rahmen der qualifizierten elektronischen Signatur geeignet sind." see also `https://www.bundesnetzagentur.de/DE/Service-Funktionen/ElektronischeVertrauensdienste/QES/WelcheAufgabenhatdieBundesnetzagentur/GeeigneteAlgorithmenfestlegen/geeignetealgorithmenfestlegen_node.html` [last accessed: Jul. 2017].

[270] Translated from the German 'Algorithmenkatalog', which is used as a short name to refer to the document.

[271] In the choice of algorithms and their technical details the BNetzA is referring the reader for details to national and international standards (e.g., DIN, BSI, NIST, ANSI, IEEE, ISO) as well as scientific publications. This is the reason why one future non-academic activity to push malleable signatures towards practical applicability is to standardise their properties, algorithms and cryptographic constructions.

The list contains the following hash-functions: SHA-256, SHA-512/256, SHA-384, SHA-512. A mechanism using those is expected to increase the evidence till the end of 2021. As digital signature algorithms it lists RSA-based algorithms, DSA as described in [355], as well as DSA variants based on elliptic curve cryptography (ECC).

This thesis uses RSASSA-PSS from PKCS-v2.2 [422] and other RSA-based algorithms as an example of being legally recognised. In the following the thesis will shortly give the reasons for this: In the 2017 edition of the list — dated 7th of December 2016 — [85], the newer RSASSA-PSS algorithms in PKCS#1 v2.2 [422] are listed as suitable mechanisms till 2022. This is under the constraint that it is operated with the recommended hash algorithms and long enough hash- respective key-lengths. Thus, the signed electronic documents will be awarded with an increased probative value until at least the year 2022[272]. Note, signatures with older algorithms might still be legally recognised today if created in the past; the algorithms then get phased out[273]. For example: Signatures generated with schemes from the PKCS#1 standard in version 1.5, like RSASSA-PKCS-v1_5, shall only be recognised as qualified electronic signatures if the algorithms and methods were used to create the signature before the end of the year 2016 [84].[274] As the SigVO has demanded the catalogue to list algorithms with a projected future suitability of six years[275], signatures with RSASSA-PKCS-v1_5, if created before 1st of January 2017, awards signed electronic documents with an increased probative value until the year 2020[276]. Also recently in January 2018, there were no major changes regarding the signature schemes observable in the related, yearly published report BSI TR-02102 [76] issued by German government agency for information security[277].

The catalogue of algorithms by BNetzA does neither explicitly endorse nor explicitly forbid the use of secure primitives in other schemes, such as malleable signatures. The catalogue of algorithms by BNetzA mentions several cryptographic primitives as secure and also several schemes. However, it does not explicitly exclude any non listed from being generally secure or endorsable in the future. One example case in Germany[a] shows that there is still some freedom on how the primitives and schemes listed in the catalogue can be applied to documents while still being legally regarded to generate a valid electronic signature: The so-called ‘container signature’ is a legally recognised electronic signature scheme that gets applied to a set of documents (see Sec. 4.5.2 for some details). While no legal case that discusses malleable signatures is known, the above case shows that it is possible to gain legal acceptance without the scheme being listed in the catalogue of algorithms issued by BNetzA. However, this requires a case by case decision and is thus such an unlisted algorithm is not suitable to provide the Verifier with legal certainty[b].

[a] BGH Beschluss vom 14. Mai 2013 – VI ZB 7/13 [81]

[b] See Definition 111 on page 153 for the understanding of this thesis of the term.

5.3.2. EU: Algorithms, key size and parameters report by ENISA

Compared to the way the German catalogue of algorithms is linked to the SigVO, the ENISA report is not directly referenced in or related to legal texts. However, this thesis treats it as an important reference as the ENISA is the organisation that advises the European Union on topics in the IT security realm. Regarding the example of RSA-based algorithms, the 2014 report [188], dated November 2014, notes that RSASSA-PSS comprises two cryptographic primitives the RSA signature and a hash algorithm. For each primitive the report lists which algorithms and parameters are considered secure. Further details can be found in the report [188].

Interestingly for this thesis, a note in ENISA’s report states:

[272] Algorithms that are deemed suitable are projected to be usable for seven years instead of the legally required six

[273] If foreseeable this is announced early on, e.g. RSASSA-PKCS-v1_5 phase out was announced in the catalogue published at the end of 2015 [84] and thus is no longer listed in the version published end of 2016 [85].

[274] “RSA-Signaturen können gemäß dem PKCS#1v1.5-Standard noch bis 2016 erstellt werden. Ihr Beweiswert bleibt bis 2020 erhalten.” [84].

[275] Anlage 1 Abschnitt 1 Nr. 2 SigVO.

[276] See footnote 274.

[277] German Federal Office for Information Security (BSI).

> "This document does not consider any mechanisms which are currently only of academic interest. In particular all the mechanisms we discuss have been standardized to some extent, and have either been deployed, or are slated to be deployed, in real systems."[278] [188]

This means that to increase the likelihood to be mentioned explicitly in ENISA's report, new methods, like malleable signature schemes, need to gain practical importance or get standardised — or both.

5.3.3. United States of America: Special publication 800-131A by NIST

Compared to the way the German catalogue of algorithms is linked to the SigVO, the publication 800-131A [25] by the National Institute of Standards and Technology (NIST) has a comparable link to US legislation. As the title 'Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths' [25] suggests it contains recommendations — similar to those of BNetzA's catalogue of algorithms or Algorithms, key size and parameters report' by ENISA — regarding the cryptographic algorithms, their recommended modes of operation and recommendations regarding the length of the keys to use them securely.

5.3.4. Digital signatures by *Goldwasser, Micali,* and *Rivest*

Goldwasser, Micali, and Rivest defined security goals for digital signatures including an adversary model and the attacks in 1988 [211].

The strongest security notion is to withstand the most sophisticated adversary that only needs to mount an attack with the lowest severity without being detected. For digital signatures they defined the attack of *existential forgery under an adaptive chosen message attacks (EF-CMA)*. The attack succeeds when adversary can produce a breach of the lowest severity, which "may only be a minor nuisance" [211]. They termed this an *existential forgery* and defined it as follows:

Definition 93 : Existential forgery by *Goldwasser et al.*

> *"Forge a signature for at least one message. The enemy has no control over the message whose signature he obtains, so it may be random or nonsensical." [211]*

In other words, such a forgery does only need to exist; the message for which a signature was forged does not need to be a sensible message.

The adversary is sophisticated as it is not constrained in its choice of the attacking message and it can see valid signatures for arbitrary messages of its own choosing. This ability is termed *adaptive chosen message* and defined as follows:

Definition 94 : Adaptive chosen message attack by *Goldwasser et al.*

> *"[...] the enemy is also allowed to use [the* Signer[279]*] $\mathcal{A}$ as an "oracle"; not only may he request from $\mathcal{A}$ signatures of messages which depend on $\mathcal{A}$'s public key but he may also request signatures of messages which depend additionally on previously obtained signatures." [211]*

The above means that the adversary can ask the original signer (denoted in the above definitions as $\mathcal{A}$) to generate a signature over a message that the adversary can freely choose. This is known as signing oracle. The adversary can use it, but will only see the resulting signature. If the signature scheme is secure then from using the oracle to generate a polynomially bounded amount of messages, the adversary shall gain no access to the secret signature key that corresponds to the public key that was fixed in the beginning. Further, the adversary can adaptively choose the next message to give to the oracle depending on previous messages and signatures obtained. Hence the name adaptive chosen message attack. This is captured in this thesis as follows (was already presented in Sec. 2.9.4):

[278] The American English spelling from original work is retained for terms and quotes.

[279] Note, originally $\mathcal{A}$ is used in this definition and it denotes the user whose signature method is being attacked, which this thesis calls Signer.

Definition 17 : Existential unforgeability under an adaptive chosen message attack following *Goldwasser et al.*

The adversary can use a signing oracle to generate valid signatures on messages of the adversary's choice. To succeed and break unforgeability, the adversary needs to generate a fresh message that verifies under a previously fixed public verification key with non-negligible probability. A message is fresh if the adversary never queried the signing oracle for this message.

Note, the forged message does not need to make sense to the application that uses these messages.

Goldwasser, Micali, and Rivest's notion can be captured formally, using the algorithms introduced in Sec. 2.9.2 and the notation used in this thesis, as follows:

Definition 95 : Existential Unforgeability under Chosen Message Attack (EUF-CMA) following *Goldwasser et al.*

A signature scheme S *consisting of three algorithms with polynomial runtime (PPT)* $\mathsf{S} = (\mathsf{KGen}_{sig}, \mathsf{Sign}, \mathsf{Verify})$ *is said to be existentially unforgeability under chosen message attack (EUF-CMA), if for any efficient (PPT) adversary* $\mathcal{A}$ *the probability that the experiment* $\mathsf{Unforgeability}_{\mathcal{A}}^{EF-CMA}(\lambda)$ *given in Fig. 28 returns* 1*, is negligible (as a function of* λ*).*

```
Experiment $\mathsf{Unforgeability}_{\mathcal{A}}^{EF-CMA}(\lambda)$
  $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^{\lambda})$
  $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(\mathsf{pk})$
    for $i = 1, 2, \ldots q$ let $m_i$ and $\sigma_i$
        denote the adaptive queries and answers to and from the oracle $\mathsf{Sign}$
  return 1 iff
    $\mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = 1$ and
    $\forall i, 1 \leq i \leq q : m^* \neq m_i$
```

Figure 28. Existential unforgeability under chosen message attack (EUF-CMA) Experiment

***Goldwasser et al.*'s EUF-CMA security for a digital signature scheme explicitly forbids authorized modifications.** EUF-CMA secure digital signatures detect forgeries of the signed message done by an adversary who has no access to the secret used to generate the signature. Thus, the security notion of EF-CMA ensures that nobody other than the Signer can produce a valid signature with respect to the Signer's public key for a new and fresh message. Under the definition of EF-CMA a Sanitizer that carries out an authorized modification forges the signature, because the Sanitizer has not used the secret signature creation key ($\mathsf{sk}_{\mathsf{sig}}$) to create a verifying signature for a new message, obtained by applying only authorized modifications, that was not signed. This also explicitly requires an EF-CMA-secure mechanism to detect any subsequent modification that did not involve the $\mathsf{sk}_{\mathsf{sig}}$.

The model of EF-CMA can only be directly applied for authorized modifications by an additional trusted party like the Sanitizer, if the Sanitizer's role is merged with the Signer's role. However, that contradicts the merit of the malleable signature schemes that introduce the Sanitizer as a new and separate role.

5.3.5. Definition of digital signatures from ITU X.800 and ISO 7498-2

Note, that ISO 7498-2 and ITU X.800 are technically very close.

Definition 96 : Digital signature mechanisms from ITU X.800

"These mechanisms define two procedures:

a) signing a data unit, and

b) verifying a signed data unit.

> *The first process uses information which is private (i.e. unique and confidential) to the signer. The second process uses procedures and information which are publicly available but from which the signer's private information cannot be deduced.*
>
> *[...] The essential characteristic of the signature mechanism is that the* ***signature can only be produced using the signer's private information****."*[280] *[263]*

ITU X.800 further elaborates that "[...] the signed data unit cannot be created by any individual except the holder of the private key [...]" [263]. This leads to the analysis that creating valid signatures by a trusted third party like the Sanitizer, which is different than the Signer — the "holder of the private key" [263] —, is not in ITU X.800's scope.

ITU X.800's digital signature explicitly forbids subsequent authorized modifications as valid signatures can only be produced using the Signer's secret key. ITU X.800 explicitly requires that the production of valid signatures involves the signer's secret signature generation key[a]. Hence, ITU X.800 does explicitly forbid subsequent authorized modifications by parties that are not in the role of the Signer.

Regarding detection of authorized subsequent modifications, ITU X.800 describes that "[...] it can subsequently be proven to a third party (e.g., a judge or arbitrator) at any time that only the unique holder of the private information could have produced the signature" [263]. The ITU X.800's notion of digital signature does not further elaborate on detection in the section defining digital signatures. However, ITU X.800 lists digital signatures as means to protect integrity, e.g. "connection integrity" [263, table 1] which means it "detects any modification" [263].

[a] "The signing process involves [...] the production of a cryptographic checkvalue of the data unit, using the signer's private information as a private key." [263]

This thesis found no difference in the definition of digital signatures in ITU X.800 compared to ISO 7498-2. The following definition is in accordance with the origin authentication from the same standard (see analysis in Sec. 5.2.1).

Definition 97 : Digital signature from ISO 7498-2

> *"Data appended to, or a cryptographic transformation of, a data unit that allows the recipient of the data unit to prove the source and integrity of the data unit and protect against forgery e.g. by the recipient" [239, p. 3]*

That same definition is also found in ISO 13888-1:2004. Like ITU X.800, the mechanism's description mandates that "the signing process involves [...] using the signer's private information as the private key." [239, p. 6] On verification it establishes that "the [valid] signature can **only** be produced using the signer's private information"[280] [239, p. 6].

ISO 7498-2 secure digital signature scheme explicitly forbid authorized modifications as production of a valid signature always requires the Signer's secret key. ISO 7498-2 definition makes a clear statement that digital signatures need to "protect against forgery" [239] by the recipient[a]. It mandates that "only" [239, p. 6] the signer's secret signature generation key $\mathsf{sk}_{\mathsf{sig}}$ can produce valid signatures. Hence, ISO 7498-2 explicitly does not allow subsequent authorized modifications by parties not in the possession of $\mathsf{sk}_{\mathsf{sig}}$, i.e., forbids valid signature created by a Sanitizer.

[a] This property cannot be achieved by symmetric key based mechanisms, but requires class 1 mechanisms as defined in ISO 10181-6 [340] and discussed in Sec. 5.1.10

5.3.6. Definition of digital signature from NIST

The US National Institute of Standards and Technology (NIST) has released a standard defining "the digital standard used to ensure the integrity of electronic documents, as well as the identity of the signer."[281]

[280] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[281] Taken from an announcement by NIST `http://www.nist.gov/itl/csd/fips-072313.cfm` [last accessed: Dec. 2015].

Definition 98 : Digital signature mechanisms from NIST FIPS 186-4

> *“A digital signature is an electronic analogue of a written signature; the digital signature can be used to provide assurance that the claimed signatory signed the information. In addition, a digital signature may be used to detect whether or not the information was modified after it was signed (i.e., to detect the integrity of the signed data).” [355]*
>
> *and*
>
> *“The result of a cryptographic transformation of data that, when properly implemented, provides a mechanism for verifying origin authentication, data integrity and signatory non-repudiation.” [355]*

In FIPS 186-4, the protection goals of a digital signature are specified: “Digital signatures are [...] to detect unauthorized modifications to data and to authenticate the identity of the signatory [...]” [355] and it shall be possible to “[...] use a digital signature as evidence in demonstrating to a third party that the signature was, in fact, generated by the claimed signatory” [355].

Thus, FIPS 186-4 sees that digital signatures are providing origin authentication, data integrity and signatory non-repudiation. The non-repudiation property is defined with a focus on classic digital signatures as follows:

Definition 99 : Non-repudiation from NIST FIPS 186-4

> *“A service that is used to provide assurance of the integrity and origin of data in such a way that the integrity and origin can be verified and validated by a third party as having originated from a specific entity in possession of the private key [...]” [355]*

FIPS 186-4 explicitly requires the DSS to detect any subsequent modification to be detected. It implicitly allows authorized modifications and requests the provision of non-repudiation in respect to the private key. FIPS 186-4 makes a clear statement that digital signatures need to “detect unauthorized modifications” [355]. Thereby it implicitly tolerates authorized modifications.
Regarding the accountability note that the term “signatory non-repudiation” [355] is focussed on classic DSS and results in the following: In case authorized modifications do not require an additional secret key for the Sanitizer this means that the Signer is the only entity that is “in possession of the private key”. Hence, following FIPS 186-4 the Signer cannot deny being accountable for any authorized modifications.

5.3.7. Notion of transferability for digital signatures

In their book, *Katz and Lindell* discuss that digital signatures in comparison with message authentication codes — that are not based on asymmetric cryptography — have the “[...] qualitative advantage that digital signatures [...] are publicly verifiable.” [278]

They describe the notion of *publicly verifiable* as follows:

Definition 100 : Public Verifiability following *Katz and Lindell*

> *A signature scheme is publicly verifiable if once a Verifier “[...] verifies the signature on a given message as being legitimate, then it is assured that all other parties who receive the signed message will also verify it as legitimate.” [278] Assuming that the other parties will have and use the same public verification key(s) as the first Verifier.*

“Public verifiability implies that signatures are transferable [...]” [278]. *Katz and Lindell* define and explain the notion of transferable as follows:

Definition 101 : Transferable signatures following *Katz and Lindell*

> *Signatures are transferable when “[...] a signature σ on a message m by a particular signer S can be shown to a third party, who can then verify herself that σ is a legitimate signature on m with respect to S’s public key ([...] this third party also knows S’s public key). By making a copy of the signature, this third party can then show the signature to another third party and convince them that S authenticated m [...]” [278]*

The notion of publicly verifiable and thus transferable signatures defined by *Katz and Lindell* is helpful for the public form of accountability.
Note, the opposite — non-transferability — has also been studied, e.g. [120, 456].

5.4. Definitions of integrity in selected existing MSS literature

None of the fundamental works on MSS define the notion of integrity in the light of authorized modifications. Namely, there are no explicit definitions of the changed integrity protection offered by MSS mentioned in [12] from *Ateniese, Chou, de Medeiros, and Tsudik*, in [349] and [347] authored by *Miyazaki* and others, in [455] by *Steinfeld, Bull, and Zheng*, nor in [273] written by *Johnson, Molnar, Song, and Wagner*.

While some of the original works contain the term 'integrity' (works by *Miyazaki* or *Johnson et al.*), the term is not re-defined in the presence of the changed protection offered by allowing to authorize subsequent modifications. Later work in the field by *Agrawal et al.* [4] attempts to describe what integrity is offered, but still the notion described has limitations. This thesis revisits the three definitions that contain the term 'integrity' in this section and shortly highlights their limitations in respect to the explicit discussion of authorized modifications.

5.4.1. Integrity by *Miyazaki*

In [349] and [347] authored by *Miyazaki* and others, the authors position the protection offered by MSS between the security goals of confidentiality and integrity following classical computer security understandings of integrity. The authors state that MSS are solving the problem that "[c]urrent digital signature schemes thus cannot assure both the integrity and the confidentiality of a document." [347, 349]. "Appropriate alteration of some signed documents, however, should be allowed because there are security requirements other than the integrity of the document." [347, 349]. This statement is found in both, [347] and [349]. No further attempts to define or describe the achieved protection regarding integrity is made in the works [347, 349]. In summary, the authors make conflicting statements: First, they state that integrity should be possibly preserved by the use of MSS. Second, they state that allowing subsequent modifications does not preserve integrity. Thus, this thesis does not find a consistent and well defined notion of integrity in the two works, neither in [349] nor in [347].

5.4.2. Integrity by *Johnson et al.*

Mentioning the word 'integrity' RSS are introduced by *Johnson, Molnar, Song, and Wagner* as follows: "Redactable signatures are intended to model a situation where a censor can delete certain substrings of a signed document without destroying the ability of the recipient to verify the integrity of the resulting (redacted) document." [273] No further definitions or descriptions of the achieved protection regarding integrity are given in [273]. The authors' notion of integrity works if it is seen as an individual property of each block: A redaction of blocks will not destroy the ability of the Verifier to "[...] verify the integrity of the resulting (redacted) document [...]" [273] which contains only remaining blocks. This however makes no statement about the protection offered to the document that still has the potential for future subsequent modifications.

5.4.3. Integrity by *Agrawal et al.*

Agrawal, Kumar, Shareef, and Rangan in [4] describe what is protected by a sanitizable signature as follows:

Definition 102 : Integrity by *Agrawal et al.*

> *"The verifier confirms the integrity of disclosed parts of the sanitized document from the signature and sanitized document." [4]*

The above description of the notion of integrity is already very precisely worded and thus marked as a definition. However, it is incomplete regarding a document that still has the potential for future subsequent modifications, because it makes no statement about the blocks which have not been subject to sanitization, i.e. "disclosed" [4] blocks. Definition 102 does not inspect the block subjected to subsequent authorized modifications nor does it contemplate the impact of differentiating between disclosed and non-disclosed blocks. It is the goal of this thesis to capture the above-mentioned details more precisely and still be able to also subsume other existing notions as special cases.

5.5. Analysis results of existing technical integrity and authenticity definitions with respect to authorized modifications

This section compiles in condensed and summarised form the intermediate findings and results obtained from this section's analysis of the existing technical definitions. The often only subtle differences in existing definitions are not surprising because many of these definitions also have a slightly different scope or focus, e.g. messages exchanged over computer networks vs. general IT security. Together with the results from the legal analysis results (Chapter 4), the results presented in this section form the basis for the thesis's definition of integrity as presented in the following chapter (Chapter 6).

5.5.1. Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law

Following ISO 2700x's view, integrity indicates the data's accuracy and completeness (see Sec. 5.1.4). But in order to know when data is complete or accurate there needs to be a defined purpose. Data contained in messages (or documents[282]) are at some point interpreted by higher level application logic or by a human, i.e. data is processed. The purpose then defines what subsequent modifications are irrelevant because their meaning is the same as the original unmodified data. It is usual that at this higher level the parsers allow what subsequent modifications to ignore, and thus define by their behaviour the equivalence of data, and finally when a modification induces a different consequence. So from the data's point of view, what constitutes a modification that is able to inflict consequences is firmly decided only by the process that processes the data. Examples of these technical harmonisation rules are the folding and unfolding of headers in message formats for the Internet, i.e., originally in RFC 0822 [129] from 1982 and updated by follow-up RFCs on the Internet Message Format, or canonicalization methods for XML documents, i.e. C14N [61]. Note that RFC 4949 even acknowledges that the "[...] ability [...] to detect changes is limited by the technology of the mechanisms used to implement the [integrity protection] service." [438].

When looking at MSS literature the initial work of *Steinfeld, Bull, and Zheng* [455] in the field of redactable signatures has already stated that "[...] it is clear that in this model the signer must have full control [...] to force some portions as manadatory [sic] for inclusion in any extracted subdocument. And it may be necessary to protect against changes in meaning of extracted portions due to certain deletions." [455]. In their work the policy is called "Content Extraction Access Structure" [455] and "[t]he signer uses this [policy] to specify which subdocuments the signer is allowing signatures to be extracted for [...]" [455]. This thesis denotes this as ADM (see Sec. 3.5 for the definition). Finally note that *Steinfeld et al.* [455] already noted that such a policy is required to correctly define the property of unforgeability as the policy "[...] leads to the [...] unforgeability requirement for a CES [content extraction signature] [...] , where the attacker can query a CES signing oracle on documents of the attacker's choice. Note that any document M queried by the attacker to the signing oracle is accompanied by a corresponding CEAS (also under the attacker's choice) specifying allowed extracted subdocuments." [457].

Following *Bishop*, this thesis further assumes that "[...] a security policy identifies authorized ways in which information may be altered and entities authorized to alter it [...]" [51] exists (see Sec. 5.1.9). This integrity policy is defined when the integrity protection is applied and must be taken into account during integrity verification. Further, this thesis sees three options to describe the application's detection requirements:

[282] The terms 'message' and 'document' are used synonymously here.

1. an adjustable integrity policy can tailor the mechanism, or
2. the policy is codified into the mechanism's design and cannot be changed, or
3. a mixture of both.

The first option makes it very explicit, which integrity policy is going to be applied. The second option is in line with RFC 4949, which states that detection "[...] is limited by the technology of the mechanisms used [...]" [438] (see Sec. 5.1.6). With the second option, the limitation of the mechanisms is not an unknown bug, but known before the mechanism is applied with consent as the protection. Hence, this thesis sees the limitation as part of an explicitly endorsed integrity policy. So with the second option, depending on the desired integrity policy a dedicated mechanisms for protection is chosen. The third option is a mixture of the first and the second and might make it harder to understand what integrity policy is going to be finally enforceable.

Together the chosen mechanism and its configuration constitute the integrity policy. This integrity policy technically defines if and what subsequent modifications are detectable and which subsequent modification are authorized and thus result in one or more documents that have the same integrity and authenticity as the original, but might not be equivalent to the original. A coverage of all information relevant to the law must be assumed, as technically enforcing that all relevant information is covered is out of scope of the technical integrity protection mechanisms. This thesis' assumption is that the limitations of mechanisms are consented to by the signatory due to the mechanism being chosen with the knowledge of its limitations. Furthermore, this thesis assumes that the configuration of the chosen mechanism — if this is possible for the chosen mechanism — is consented to when the signatory specifies the configuration. In combination the two choices constitute the integrity policy.

Whether or not an action on data is considered a modification of that data depends on the further processing of the data by applications — or finally by humans. While the technical equivalence of documents is defined within the technical mechanisms and thus is in scope of this thesis, identifying what legally constitutes information that requires protection due to it being of relevance in law[a] is out of scope. This thesis assumes that the signatory takes care that all the data that contains information that is relevant in law is getting correctly protected. For example it might not be relevant in law if text is copied from a black text into a dark-grey text as long as it is still clearly legible to the human, but it might be relevant if the text's content remained unchanged.

Hence, integrity depends on the application-specific building of equivalence classes. Which subsequently modified documents are still in the same class as the original, i.e., have the same integrity and authenticity as the original, but might not be equivalent in every aspect to the original. Following *Biba* (see Sec. 5.1.1), this thesis assumes that "[...] integrity does not imply guarantees concerning the absolute behaviour of systems [...]" [47, p. 9]. The application-centric view of this thesis is also in agreement with *Gollmann*'s statement that a distinction if data is false "on purpose" or "false by accident" [215] "does not matter for the application" [215] (see Sec. 5.2.6). Hence, all modifications that do not adversely affect the application are considered authorized. Then all the documents that have been modified in a way that does not adversely affect the application are considered in the same equivalence class.

[a] 'Relevance of law' translated from the German notion of 'rechtserheblich' using [152].

The following example highlights this application-dependence and the resulting equivalence classes and their members are exemplified and put in a Venn-style diagram (see Fig. 30).

EXAMPLE

Example: Assume the application is an audio CD player. In other words, the message is the data on the audio CD and the process or application is the rendering of that data into an audio signal to allow listening to the music from the CD. The output, i.e. the music one can listen to, from an error-free copy of an audio CD is as good as listening to the original pressed audio CD.

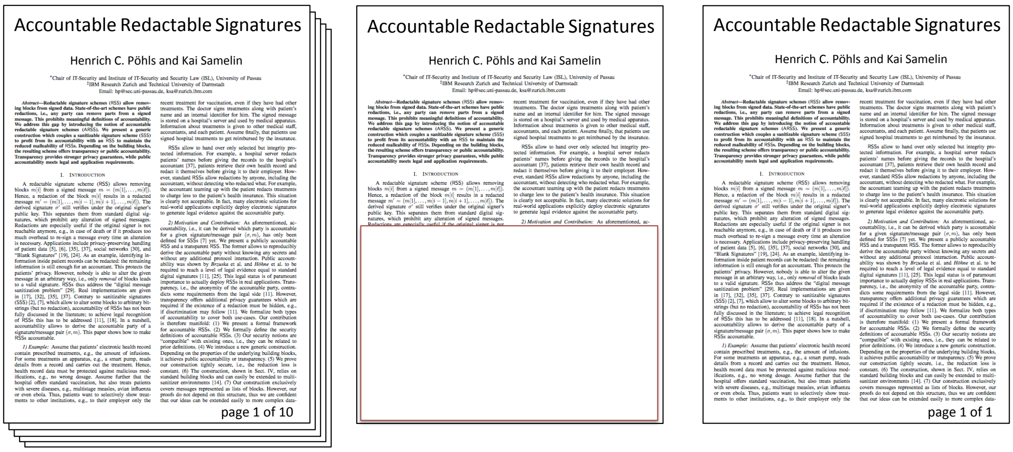
For integrity, a duplication of data — that is a copy done without modifying any interesting details of the original data — does not impede integrity as the copy of the data is not modified in ways that impede the usability of data as the obtainable information from the copy would be fit for the purpose. Hence, in the Venn-style diagram Fig. 30, the `Copy` is not in the set of modifications, but still considered unmodified, as there is no observable difference when playing it. In other words, the modification of copying data exactly is not harmful for the integrity of the audio CD player application. However, this view changes when the application changes: Consider that making an unmodified copy destroys a property like the uniqueness of the original that a digital rights management (DRM) application might want to protect.

To further support that the notion of modification is application-dependent consider another — constructed — example.

EXAMPLE

Example: Assume the purpose of the application is to identify author(s) and title of a multipage document. Assume further that the list of authors and the titles are always contained solely on the upper half of the first page of all multipage documents. The application facilitates a parser which only parses the first page of a given multipage document, depicted in the left of Fig. 29. During parsing, the application identifies authors, title and the total number of pages. Assume that the total number of pages is always and solely given in the lower right corner through a statement like 'page X of Y'. The parser's output as well as the resulting application's decision on the title and author are considered observable behaviour in this example. However, the information about the total number of pages is considered only optional output of the parser from the point of the application. It suffices for the application's expected behaviour that this intermediate information is not correctly gathered or not gathered at all.

Consider the following changes to the original document used as input to the application:


Accountable Redactable Signatures

Henrich C. Pöhls and Kai Samelin

page 1 of 10

Accountable Redactable Signatures

Henrich C. Pöhls and Kai Samelin

Accountable Redactable Signatures

Henrich C. Pöhls and Kai Samelin

page 1 of 1

Figure 29. Left: Original; Middle: `Only_upper_half_of_1st_page_copied`, title and authors still preserved but lost total number of pages; Right: `Only_1st_-page_pagenumbers_adjusted`, title and authors still preserved but lost total number of pages

- `1st_page_copied` will retain just the first page. The parser would be able to produce the same results if it would work on the first page of the original unmodified multipage document, or if it would work on a document that contained only the first page of the original document as an exact copy. The application's purpose is not affected and thus no adverse affect on the application-dependent integrity can be noted. Further, no observable differences are detected, i.e. the parser's output is the same. Thus, removing the rest of the document for this application is **not a modification**. Thus, this modification is considered authorized. In the Venn-style diagram Fig. 30 the `1st_page_copied` is not a modification.
- `Only_upper_half_of_1st_page_copied` is a modification which retains only the upper half of the first half page in the document and removes the rest. This would still support identifying authors and title, but it results in a different output of the parser when compared to the original as it prohibits parsing the total page count correctly. Thus, this modification is considered authorized as it does not adversely affect the application-dependent integrity. Fig. 30 notes this as an authorized modification.

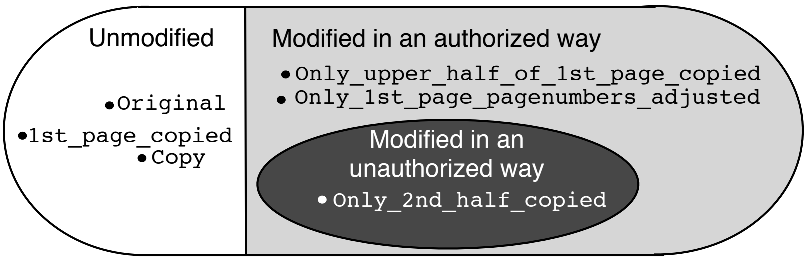

Unmodified
•Original
•1st_page_copied
•Copy
Modified in an authorized way
•Only_upper_half_of_1st_page_copied
•Only_1st_page_pagenumbers_adjusted
Modified in an unauthorized way
•Only_2nd_half_copied

Figure 30. Venn-style diagram of dividing the set of all modifications into subsets of 'unmodified', 'modified in an unauthorized way' and 'modified in an authorized way'; modifications and original from the example are indicated by a • as elements of those sets. Note, Fig. 31 shows the final tri-section.

- `Only_1st_page_pagenumbers_adjusted` is a modification which retains the contents of the first page but adjusts the total page count to one. As above, the parser's output will look slightly different compared to the output from the original as it would not be able to get the correct total number of pages. The modification still allows to support the application identifying authors and title. However, the modifications result in different parser output regarding the page count being just one. Thus, this modification is considered authorized as it does not adversely affect the application-dependent integrity. Fig. 30 notes it as an authorized modification.
- `Only_2nd_half_copied` would be a modification that results in only the second half of the document's pages remaining, the first half of all pages are deleted. After this, the parser will not be able to identify title and authors, as this information was assumed to be solely parseable from the first page. This adversely affects the behaviour of the application. Thus, this is an unauthorized modification, as marked in Fig. 30.

This example concludes Analysis Result 8[283] finding that the decision what constitutes a modification can only be made depending on the application. Furthermore, the equivalence of documents gets technically defined by the message formats, their implementation in parsers and the integrity protection mechanism in its actual configuration defines the members in each equivalence class. The classes are unmodified vs. modified.

5.5.2. Analysis Result 9: Definition of authorized and unauthorized modification; and modification in general

Analysis Result 8[283] identified that the decision in which class a changed document is in respect to the original is application-dependent, i.e., if the change is actually a modification depends on the application. This thesis considers an application that behaves observably different due to a changed input as the deciding technical aspect. The application can only behave differently if the parsers or the integrity verification in its actual configuration detects the change. This generally defines the changes that makes the documents members in either one of the two equivalence classes of *unmodified* and *modified*. In reverse further differentiation of these observable differences then allows to further re-define the class of modified documents into two: The class of modified documents where the modifications are a threat to the verifiable state of integrity and thus are unauthorized; and the class of modified documents where the modifications are identified as existent but benign and thus are deemed authorized. This thesis in the following gives definitions for authorized and unauthorized modifications and also for modifications of data in general. Overall, the three classes of unmodified, authorized and unauthorized modifications can be separated; the Venn-style diagram in Fig. 31 depicts their dichotomy.

[283] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

5.5.2.1. Definition of unauthorized modification

An unwanted and hence unauthorized modification could be inflicted on data by a party with malicious intent, but also by error.[284] This thesis defines an unauthorized modification as follows:

Definition 103 : Unauthorized Modification

> *An* ***unauthorized*** *modification of data is any alteration of data (including the creation or insertion or deletion of data) that results in* ***unauthorized*** *observable consequence(s) during further processing when compared to the further processing of unmodified data (including, but not limited to an observably different unauthorized output).*
>
> *Notes for Definition 103:*
>
> *Note 1 Definition 103 defines an unauthorized modification as application-dependent, by its reference to further processing.*
>
> *Note 2 Definition 103 makes no statement about the fitness or suitability for the further processing — neither of the original data nor of the modified data.*
>
> *Note 3 Definition 103 makes a detection of the occurrence of a modification possible due to the notion of observable consequence(s).*
>
> *Note 4 Definition 103 includes deletion of data as a modification. Depending on the definition deletion is also an attack on availability.*
>
> *Note 5 Definition 103 does not differentiate why the modification has occurred (i.e. it makes no distinction between a malicious active attacker or an error).*

The definition for unauthorized modification given in Definition 103 has been formulated to be in line with the technical integrity definition found in the standards ISO 7498-2[285] and ITU X.800[286] as analysed previously in Sec. 5.1.5.

The notes are added for clarity, they explicitly state the wanted interpretations and clarify the definition following the observations made during the analysis.

Note 1 follows from the previous analysis result (Analysis Result 8[287]), Definition 103 defines an unauthorized modification not independent of the application, i.e., it is defined with respect to "further processing".

Note 2 excludes any claims from Definition 103 that the data either before or after the modification has external consistency or can be processed further, as defined by *Clark and Wilson* [125]. Following *Gollmann*, who stated that not even the original, unmodified data (the logical asset) needs to correspond to the real world object (the physical asset) it describes or refers to (see Sec. 5.1.8), Definition 103 explicitly notes that this does — naturally — also not hold for unauthorized modified data.

Note 3 states explicitly that for Definition 103 the unauthorized change must have "observable consequence(s)", this means that at some level the modification can potentially manifest itself and it is then that it can be detected. Further, Definition 103 explicitly excludes the recovery from unauthorized modifications, i.e., it does not include ISO 10181-6 Class 4 [340]. As stated in Sec. 5.5.1, what constitutes a modification depends on the application. If the consequences are unnoticeable in the application it is not a modification. If the modification leads to detectable consequences in the further processing then it is a modification. How this deviation can be detected is out of the definition's scope. Also, whether or not a modification operation is generating authorized deviations is again subject to the application. Unauthorized modifications are a seen as a sub-set of authorized modifications as depicted in Fig. 31.

Note 4 states that the definition of unauthorized modification includes the active attack of deletion. Hence, deletion is not seen as part of the availability, but of the integrity property.

[284] Note, a distinction if data is false "on purpose" or "false by accident" [215] "does not matter for the application" [215]; see also Sec. 5.2.6.

[285] "The property that data has not been altered or destroyed in an unauthorized manner." [239, p. 3] [263].

[286] "[...] altered without detection and results in an unauthorized effect [...]" [263].

[287] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

Finally, Note 5 explicitly makes sure that the definition is not affected by the reason for a modification. This follows the statement by *Gollmann*, who noted that application's processing of modified data will generate unwanted consequences no matter what caused the modification [215] (see Sec. 5.2.6).

5.5.2.2. Definition of modification in general

Removing the weighting — and thereby ignoring if the modification was authorized or not — yields this thesis's definition of a modification in general. The definition of a modification in general is as follows:

Definition 104 : Modification (in general)

> *A modification of data is any alteration of data (including the creation or insertion or deletion of data) that results in observable consequence(s) during further processing (including, but not limited to, an observable different output) when comparing it with the processing or output of unmodified data.*
>
> *Notes for Definition 104:*
>
> *Note 1 Definition 104 defines modification as application dependent, by its reference to further processing.*
>
> *Note 2 Definition 104 makes no statement about the fitness or suitability for the further processing — neither of the original data nor of the modified data.*
>
> *Note 3 Definition 104 makes a detection of the occurrence of an unauthorized modification possible due to the notion of observable consequences.*
>
> *Note 4 Definition 104 includes deletion of data as an unauthorized modification. Depending on the definition deletion is also an attack on availability.*
>
> *Note 5 Definition 104 does not differentiate why the modification has occurred (i.e. it makes no distinction between a malicious active attacker, or error).*

The reasons for the notes in above definition are the same as those for Definition 104. Next, this definition of a modification in general is used in combination with the definition of unauthorized modification to define an authorized modification.

5.5.2.3. Definition of authorized modification

This thesis argues that "appropriate alteration" [347] or in other words *authorized modifications* exist. The argument applies the operation of set subsection as follows: authorized modifications are all those general modifications (as defined in Definition 104) that are not unauthorized modifications (as defined in Definition 103).

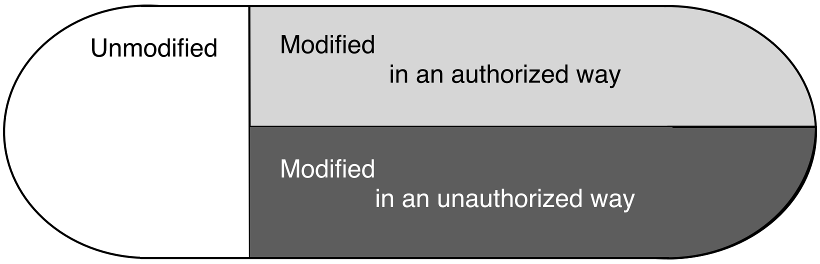

Unmodified
Modified in an authorized way
Modified in an unauthorized way

Figure 31. Venn-style diagram of differentiating documents into disjunct sets of 'unmodified' and 'modified'; Further dissecting the 'modified' subset into 'modified in an unauthorized way' and 'modified in an authorized way' subsets

By this argument, not only are authorized and unauthorized modification mutually exclusive:

$$\text{authorized\_modified} \cap \text{unauthorized\_modified} = \emptyset\,.$$

But, when authorized and unauthorized modifications are combined they account for all modifications:

$$\text{modified} \cap \{\text{authorized\_modified} \cup \text{unauthorized\_modified}\} = \emptyset\,.$$

This notion requires that for all documents it must be possible to classify them in exactly one of the three classes. This is an assumption that is in line with common sense and common security level classification: Common sense is that documents can be classified either as unmodified or as modified. If subsequent modifications cannot be detected then the document cannot be assumed to be unmodified. In security classifications the inability to acquire assurance that something meets a certain minimal level of security results in it getting classified into the lowest class, e.g. if something is unknown it is tagged as insecure. In this line, a document for which subsequent modifications cannot be detected will — for security reasons — by default be assumed to have been subject to unauthorized modifications. The Venn-style diagram in Fig. 31 depicts the outcome of the application of set subsection and the dichotomy of authorized and unauthorized modifications.

From this follows the thesis's definition for authorized modifications:

Definition 105 : Authorized Modification

*An **authorized** modification of data is any alteration of data (including the creation, insertion or deletion of data) that results **only** in **authorized** observable consequences during further processing when compared to the further processing of unmodified data (including, but not limited to an observably different **and authorized** output).*

Notes for Definition 105:

Note 1 *Definition 105 defines an **authorized** modification as application-dependent, by its reference to the further processing.*

Note 2 *Definition 105 makes no statement about the fitness or suitability for the further processing of the original data nor of the modified data.*

Note 3 *Definition 105 makes a detection of the occurrence of an **authorized** modification possible due to the notion of observable consequences.*

Note 4 *Definition 105 includes deletion of data as an **authorized** modification. Depending on the definition of availability the action of deletion is an attack on availability.*

Note 5 *Definition 105 does not differentiate why the modification has occurred (i.e., it makes no distinction between a malicious active attacker, **an authorized third party** or an error).*

The reasons for the notes in above definition are the same as those for Definition 103.

5.5.3. Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification

Nearly all analysed technical definitions of integrity state that an **unauthorized** modification must be detected to protect the integrity of data: The detection of any subsequent unauthorized modification to integrity-protected data by the Verifier is the basic requirement[288]. The exception are definitions that only consider preventative integrity mechanisms, as defined as Class 4 in ISO 10181-6. Namely, the definition by *Clark and Wilson*. If an unauthorized modification is detected, the level of detection might vary. For example detection can vary in the scope: Following *Bishop* detection might "report [...] a specific part" or "report [...] the file" [51, p. 5] (see Sec. 5.1.9). The following definitions and notions of integrity require to detect any subsequent modification to data: ITU X.800, ISO 7498-2, RFC 4949, *Stallings*, *Gollmann* (see Sec. 5.1.8), EF-CMA security by *Goldwasser et al.*. This is denoted in the last column of Tab. 1, which has the title 'explicitly require detecting any subsequent modifications'.

[288] Additional functionality might be to allow recovery, but those functionalities are not further needed in the course of this thesis.

5.5.4. Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect

First, the results for general technical definitions are presented. An overview is presented in Tab. 1. Second, the results of the analysed definitions from MSS literature (see Sec. 5.4) are given.

Definition	**explicitly forbid** authorized modifications	**explicitly allow** authorized modifications	**implicitly tolerate** authorized modifications	**explicitly** require detecting **any subsequent** modifications
Integrity Considerations of Secure Computer Systems by *Biba*	—	allow[289]	—	no
Data Integrity by *TCSEC*	—	—	tolerate[290]	no
Data Integrity by *Clark and Wilson*	—	allow[291]	—	—[292]
Integrity by *ISO 2700x*	—	—	tolerate[293]	no
Data Integrity by ITU X.800	—	—	tolerate[294]	yes
Data Integrity by *ISO 7498-2*	—	—	tolerate[295]	yes
Data Integrity from RFC 4949	—	—	tolerate[296]	yes
Integrity of Network Messages by *Stallings*	forbid[297]	—	—	yes
Data Integrity by *Gollmann*	—	—	tolerate[298]	yes
Data Integrity by *Bishop*	—	allow[299]	—	no
Digital Signature by *ISO 7498-2*	forbid[300]	—	—	yes
Digital Signatures by ITU X.800	forbid[301]	—	—	yes
Integrity framework by ISO 10181-6	—	—	tolerate[302]	no
EF-CMA secure Digital Signatures by *Goldwasser et al.*	forbid[303]	—	—	yes
Digital Signatures by *NIST FIPS 186-4*	—	—	tolerate[304]	yes[305]

Table 1. Integrity related computer science definition: Treatment of authorized modifications and requirement to detect any subsequent modification

All technical definitions of detective integrity analysed are concerned with detecting **un**authorized modifications. The one preventative definition by *Clark and Wilson* is excluded in the remainder of this subsection's analysis due to being preventative and the focus being on detective measures[306], like digital signatures (compare also Sec. 5.5.6). Regarding **authorized** modifications, only a few technical

[289] "[...] objects may not be directly modified by subject possessing insufficient privilege [...]" [47, p. 30] and "[...] only proper modifications are made [...]" [47, p. 14].

[290] data must "[...] not been exposed to accidental or malicious alteration [...]" [308].

[291] "[...] well-formed transactions [...]" [125].

[292] This is a preventative integrity notion, meaning a system that offers data integrity according to *Clark and Wilson* has to prevent any integrity violation from happening.

[293] data must retain "[...] accuracy and completeness [...]" [249]

[294] Data must "[...] not been altered or destroyed in an unauthorized manner." [263].

[295] Data must "[...] not been altered [...] in an unauthorized manner [...]" [239].

[296] "[...] unauthorized or accidental manner [...]" [438]

[297] "[...] messages are received as sent [...]" [452].

[298] "[...] as it was supposed to be [...]" [213, 214] and "[...] detect whether a document has been changed; [...]" [213, p. 187, 2nd ed.].

[299] "[...] a security policy identifies authorized ways in which information may be altered and entities authorized to alter it [...]" [51, p. 97].

[300] "[...] signature can only be produced using the signer's private information [...]" [239].

[301] "[...] signature can only be produced using the signer's private information [...]" [263].

[302] "[...] checks integrity-protected data for modifications [...]" [340].

[303] The model requires valid signatures to be only produced with the signer's private information.

[304] "detect unauthorized modifications" [355]

[305] "detect whether or not the information was modified after it was signed" [355]

[306] "[...] there is not always a direct trade-off between prevention and detection [...]" [214] also because detective mechanisms are needed to provide sufficient evidence that the preventive integrity protection is working.

definitions for data integrity make explicit statements and allow or forbid authorized changes explicitly. Particularly, the definitions by *Biba* and by *Bishop* allow them explicitly. The definition by *Stallings*, the definition of digital signatures by ITU X.800, and the definition EF-CMA security by *Goldwasser et al.* explicitly allow or forbid any authorized subsequent modifications. The remaining eight out of thirteen technical definitions in the area of data integrity and digital signatures tolerate them by not being explicitly or implicitly ruling them out. Hence, this thesis finds that following the majority of existing notions integrity notion needs to be capable of capturing that authorized modification are allowed.

The analysed definitions of an integrity notion found in MSS paint a diverse picture:

- *Miyazaki et al.* make contradicting statements even if MSS will preserve the integrity [347, 349].
- *Johnson et al.* clearly state that integrity is not destroyed by deletion of "certain substrings" [273] in an RSS[307].
- *Agrawal et al.*'s Definition 102 does make a statement about the integrity of the contents or "disclosed parts of the sanitized document"[308], however it does not offer an integrity notion that captures the protection offered to the document containing still block(s) subject to potential future subsequent authorized modifications.

5.5.5. Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document

Following ISO 7498-2, accountability "[...] ensures that the actions of an entity may be traced uniquely to the entity." [239, 263]. Following *Gollmann*, this thesis assumes data integrity to be verifiable as a pre-requisite for origin authentication [214] (see Sec. 5.2.6). Accountability for data requires that the entity, from which the integrity-protected data originated, can be established. This thesis denotes this party with the term originator.

Definition 106 : Originator

> ***Originator*** *is the party that the verification algorithm denotes as the source of the data when evaluating the origin authentication protection inherent to the integrity protection that was initially applied.*

Further, to hold the originator, as identified by data origin authentication, accountable the mechanism needs to achieve the property of non-repudiation.

None of the analysed technical definitions explicitly forbids that a subsequent modification that was deemed authorized might not adversely affect the integrity. The ITU X.800's notion of data origin authentication is even explicitly only concerned with "source of the data unit" [263] and not with its integrity (see Sec. 5.2.2). Thus, ITU X.800 tolerates even that a subsequent modification is not being recorded in a change of accountability[309].

All analysed definitions have the following in common: The Signer is the "original source" [438] of the protected message, data or "contents" [263]. If this claimed source is not updated from Signer to Sanitizer due to a subsequent modification then the original source, e.g. the Signer, must be allowed to repudiate being the origin of those modified contents. Hence, after any subsequent modification, the accountability of the original originator is lost in all the technical definitions analysed.

Regarding the accountability of the party executing a subsequent authorized modification, the definitions in X.800, ISO 7498-2, *Gollmann*'s notion and also the data origin authentication definition found in ISO 13888-1 do not proclaim that the party doing the modification must itself become accountable. They only define that the original source, i.e. the Signer, is able to repudiate his accountability.

[307] "Redactable signatures are intended to model a situation where a censor **can delete certain substrings of a signed document without destroying** the ability of the recipient to verify **the integrity of the resulting (redacted) document**." [273]

[308] "The verifier confirms the integrity of disclosed parts of the sanitized document from the signature and sanitized document." [4]

[309] "[...] does not provide protection against [...] modification" [263]. See also footnote 312.

For non-repudiation, however, ISO 13888-1 is different to ITU X.800 and *Gollmann*: ISO 13888-1 prescribes that the mechanism for non-repudiation for data must always correctly attribute the origin. Hence, for data which is potentially subject to future authorized modifications, ISO 13888-1 requires a mechanism that correctly attributes the entity that has last done modifications, i.e., it requires to have the Sanitizer accountable once it had modified the contents, and the Signer only if the contents are unchanged. Note, here modification could also be that the Sanitizer has just touched the message, i.e. actively executed its chance for a modification without modifying any content (see Analysis Result 8, Definition 54, or Sec. 3.4.3).

Definition	**explicitly forbids** to detect Sanitizer's action(s)	**explicitly requires** to detect Sanitizer's action(s)	**implicitly requires** to detect Sanitizer's action(s)	**explicitly requires** Sanitizer's accountability for its action(s)
Data origin authentication by *ISO 7498-2* and *ISO 13888-1*	—	yes[310]	—	no
Data origin authentication by *ITU X.800*	—	—	even no detection is tolerated[311]	no
Non-repudiation with proof of origin by *ITU X.800*	—	—	tolerate[312]	no
Non-repudiation of origin by *ISO 13888-1*	—	yes[313]	—	yes
Non-repudiation of origin by *RFC 4949*	—	yes [314]	—	
Data origin authentication and non-repudiation by *Gollmann* [214, 215]	—	—	tolerate	no[315]

Table 2. Origin authentication and non-repudiation related computer science definition and their treatment of authorized modifications and requirement to make Sanitizer accountable for any subsequent modification

Tab. 2 gives a shortened overview of the analysed existing definitions. From the analysis, this thesis deducts that accountability shall include the obvious capability of data origin authentication but also the non-repudiation property as a technical 'evidence'. This thesis sees a key point of accountability that for data with a still valid integrity protection a selected entity can identify and hold accountable the entity that produced the signed data's 'current state'. This follows ISO 13888-1 and the fact that in all other analysed definitions, data origin authentication and non-repudiation are just voided if the Sanitizer cannot be held accountable. The general notion of accountability of this thesis also does not require that the Verifier can achieve this solely on the basis of publicly available information such as public keys $\mathsf{pk}_{\mathsf{sig}}$ or $\mathsf{pk}_{\mathsf{san}}$ and without any additional interaction (later defined as non-interactive public accountability, see Sec. 6.4.3). This means that generally the accountability must not be *transferrable* — in the broader sense of the notion as discussed in Sec. 5.3.7 — but this thesis allows accountability generally to be done interactively involving a third party.

This thesis defines accountability as follows:

[310] In order to always reflect correctly that the "source of data received is as claimed" [239].

[311] "[...] does not provide protection against [...] modification of data units" [263]; "[...] provide for the corroboration of the source [...]" being the "[...] claimed peer [...]" [263], but this can change after the subsequent edit and be either Sanitizer or Signer.

[312] "[...] sender to falsely deny sending the data or its contents [...]" [263]; but sender can either be Sanitizer or Signer if the Signer can be held accountable if there was no subsequent modification at all and the Sanitizer can be held accountable if it did a subsequent authorized modification.

[313] With the notion "having created the content of a message [...]" [243] the definition refers to the content which after a subsequent modification will be the Sanitizer and no longer the Signer.

[314] This is due to the linkage made to data integrity which requires to detect authorized subsequent modifications.

[315] "[...] authentication and non-repudiation verify the claimed origin of assertions [...]" [215]; by relating only to the claimed origin the definition does not explicitly need the Sanitizer to become accountable, the definition only requires that if the Signer is the claimed origin the Signer is accountable if no subsequent modification occurred, and the Sanitizer is accountable if the Sanitizer is the claimed origin and the Sanitizer did an authorized subsequent modification.

Definition 107 : Accountability

> ***Accountability** allows a predefined set of entities to determine a valid signature's origin (Signer or Sanitizer) according to a protocol, i.e., decide which party is accountable (split in Signer-accountability and Sanitizer-accountability) for the signature of a given valid message-signature pair (m, σ).*

This definition is in line with the existing technical definitions analysed in this chapter, but it is also in accordance with those definitions of accountability given for malleable signatures presented later in this thesis (see Sec. 11.6.6 for an existing accountability notion in SSS, or Sec. 13.1 and Sec. 13.2 for refined notions).

5.5.6. Analysis Result 13: Digital signature schemes are a suitable integrity and authenticity protection mechanism

This thesis focuses on digital signatures as a technical mechanism to increase the Probative value. This view is supported by a wide range of technical standardisation bodies, when they list digital signature's properties: The ISO standard for non-repudiation (ISO 13888) describes digital signatures as a technical mean to generate evidence. Further, ISO 10181-6 [340] on integrity frameworks lists digital signatures based on cryptographic primitives as suitable to protect the integrity and are suitable as evidence generating mechanism. Also, ISO 27037 [251] regarding the preservation of digital evidence suggests to use digital signatures as a technical mechanism.[316] Moreover, this recognition of standardisation bodies is backed by a great body of work regarding digital signature's cryptographic security properties. This includes cryptographic proofs for secure instantiations of digital signature schemes. Hence, this thesis defined the terms classic digital signature scheme in Definition 14 and certificate in Definition 26 taking into account existing technical digital signature definitions.

[316] "[...] seal the acquired data using verification functions or digital signatures to determine that the digital evidence copies are equivalent to the originals." [251].

6 — Proposed extended integrity definition

Overview of Chapter 6

Schneier stated that “Integrity is harder to define” [433] than other security goals and *Gollmann*’s computer security textbook acknowledges already in the first edition in 1999 that it is “quite difficult to give a concise definition of integrity” [212]. Over ten years later the situation has not changed and the 2011 edition by *Gollmann* states that it is “not easy to give a concise definition” [214] of integrity. In line with this, the previous analysis of technical and legal definitions found ambiguities within each field and between them. Additionally, the integrity-preserving state of ‘authorized modified’ that is central to the integrity provided by malleable signatures was not sufficiently describable using established computer science definitions nor existing definitions within IT law. Following the methodology[317], Chapter 6 fixes this and proposes a definition of integrity which allows to clearly differentiate and compare malleable signature schemes among each other and with classic digital signature schemes.

Following Analysis Result 11[318] this thesis found that existing definitions often lack an explicit statement on the policy that defines which changes should not invalidate a corresponding integrity protection. Further, Analysis Result 8[319] found that this is defined by application requirements and thus the integrity protection mechanism shall be able to cater for different requirements by offering flexible policies.

First, the offered new extended definition introduces the aspect of allowed malleability. This allows to explicitly state if the flexibility to authorize subsequent modifications is given reflecting the applications’ needs. Some previous integrity definitions contained this implicitly (see Analysis Result 11[318]).

Second, to argue for the probative value of data signed with malleable signature schemes, their properties need to be stated such that they are comparable to those of existing legally recognised classical digital signature schemes. The analysis from previous chapters found the existing technical and legal definitions of the term “Data Integrity” to be neither adequately, nor precisely defined in respect to authorized subsequent modifications.

The extended integrity notion proposed in this chapter introduces a legally needed distinguishing aspect of non-interactive public accountability to the integrity notion. This describes the integrity protection mechanism’s capability to offer distinguishable levels regarding the detection of the integrity violations including detecting respective positions or number of violations within the document. This is required to further discuss the notion of accountability for malleable signatures, especially based on Analysis Result 12[320]. Further, the extended definition defines which form of accountability is reached. Especially, the notion of non-interactive public accountability (PUB) — which is one main contribution of the thesis — is used to clearly capture the possibility of Verifiers to identify the accountable party without further interaction and without knowledge of secrets.

[317] This is related to step 2.1 (Define a precise enough terminology to capture the requirement) of the methodology from Sec. 1.5.

[318] Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect (see Sec. 5.5.4).

[319] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

[320] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

6.1. Integrity extended by three aspects: Allowed authorized modifications, detectability of modifications and accountability for modifications

Foremost, following Analysis Result 2[321] and Analysis Result 11[322] it was considered as a goal — in order to be beneficial for the co-existence of MSS and CDSS but also useful for general clarity — that the integrity notion captures explicitly if authorized changes are allowed or not. Thus, the **first aspect** is: Explicitly state if an authorization of subsequent modifications is possible (see Sec. 6.2).

Moreover, for an integrity mechanism it shall be possible to determine whether or not the mechanism is able to allow for the detection of any subsequent modification. Especially if the mechanisms allows to authorize such subsequent modifications. This follows from Analysis Result 4[323] and Analysis Result 10[324]. As the message might be composed from identifiable blocks this detection can be offered on the level of blocks as well as on the higher level of the message. Hence, the **second aspect** is: Explicitly state the mechanism's level of detection (see Sec. 6.3).

An integrity mechanism — more precisely the included accountability protection mechanism — needs to prove who is accountable following Analysis Result 5[325], thus it requires for a mechanism to be able to state how a subsequent authorized modification will influence the accountability. The Analysis Result 7[326] shows that alongside the delegation needs to be captured. This becomes the **third aspect**: Explicitly state for the mechanism which party becomes accountable and how accountability is established in the case a subsequent modification (authorized and unauthorized) has occurred (see Sec. 6.4).

The three aspects translate into the following questions, which a concise integrity notion shall allow answering:

1^{st} aspect: *Can subsequent modifications be authorized?*

2^{nd} aspect: *What level of detectability of subsequent modifications is offered?*

3^{rd} aspect: *Who is accountable — and who can determine who is?*

Fig. 32 gives an overview of the existing technical definitions analysed in Chapter 5 when they are grouped by the first two aspects.[327]

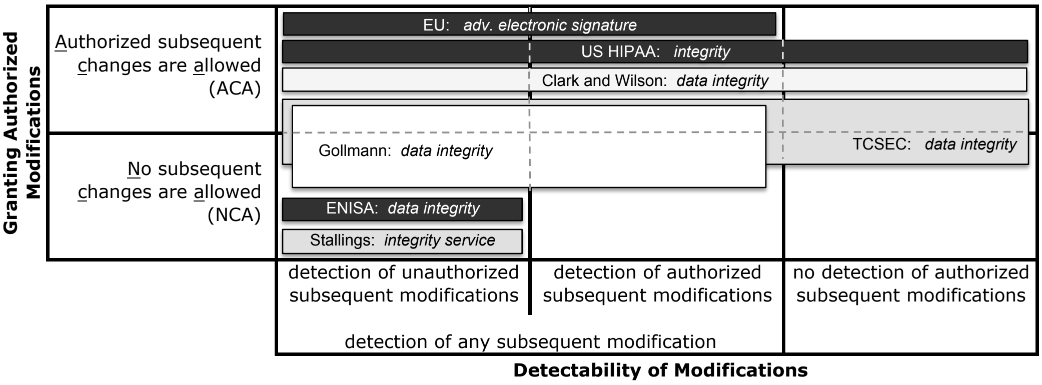

Granting Authorized Modifications
Authorized subsequent changes are allowed (ACA)
No subsequent changes are allowed (NCA)
EU: adv. electronic signature
US HIPAA: integrity
Clark and Wilson: data integrity
Gollmann: data integrity
TCSEC: data integrity
ENISA: data integrity
Stallings: integrity service
detection of unauthorized subsequent modifications
detection of authorized subsequent modifications
no detection of authorized subsequent modifications
detection of any subsequent modification
Detectability of Modifications

Figure 32. Comparison of selected integrity definitions along two aspects:
1^{st}: Can subsequent modifications be authorized?
2^{nd}: What level of detectability of subsequent modifications is offered?

[321] Analysis Result 2: Legal acts do not explicitly forbid authorized modifications (see Sec. 4.6.2 on page 101).

[322] Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect (see Sec. 5.5.4).

[323] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

[324] Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification (see Sec. 5.5.3 on page 136).

[325] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[326] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[327] Fig. 32 is an improved and leaner version of a figure that was published in [381].

6.2. 1^{st} aspect: authorized or no subsequent changes allowed (UCA, ACA, NCA)

The first aspect to introduce explicitly to the notion of integrity is a description stating whether or not authorized modifications to integrity-protected messages are granted or not. This follows Analysis Result 11[328] and Analysis Result 2[329]. Moreover, the notion of a modification as defined in Definition 104 and the notion of an authorized modification as defined in Definition 103 on page 134 is used. This follows Analysis Result 9[330].

This thesis differentiates the following three levels of increasing strictness:

- **unauthorized subsequent changes allowed (UCA):** The integrity status remains unchanged regardless of the subsequent modification. UCA offers no integrity. UCA is added to have a theoretical unique infimum; this allows the levels to be a lattice.
- **authorised subsequent changes allowed (ACA):** The integrity status remains unchanged if the applied subsequent modification to the protected message was authorized.
- **no subsequent changes allowed (NCA):** A subsequent modification to integrity-protected data results in a negative integrity status. NCA is the theoretical unique supremum; this allows the levels to be a lattice.

See Fig. 33 for a visual representation of the lattice. The levels ACA and NCA are of practical importance as they allow protecting integrity. Note, UCA offers no integrity protection.

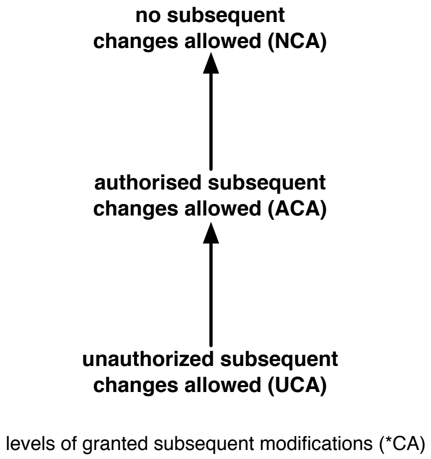

no subsequent changes allowed (NCA)
authorised subsequent changes allowed (ACA)
unauthorized subsequent changes allowed (UCA)
levels of granted subsequent modifications (*CA)

Figure 33. Lattice of the three levels of increasing strictness for tolerable subsequent modifications

6.2.1. Unauthorized subsequent changes allowed (UCA)

UCA does not offer cryptographic integrity protection. It is a theoretical low and will not be further considered in this thesis.

6.2.2. No subsequent changes allowed (NCA)

NCA covers the classic digital signature scheme (CDSS) where a combination of a digital signature scheme with a cryptographic hash-function is used to sign the hash-value of the message, which allows to detect any subsequent modification of a bit in the message with overwhelming probability. This is on the one hand due to the cryptographic properties of the hash-function. On the other hand, this is due to the nature of the involved asymmetric cryptographic operation, which does not allow to reproduce a valid signature over an adjusted hash-value without access to the private signature generation key.

[328] Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect (see Sec. 5.5.4).

[329] Analysis Result 2: Legal acts do not explicitly forbid authorized modifications (see Sec. 4.6.2 on page 101).

[330] Analysis Result 9: Definition of authorized and unauthorized modification; and modification in general (see Sec. 5.5.2 on page 133).

6.2.3. Authorized subsequent changes allowed (ACA)

In general, ACA allows to describe a very broad authorization possibility. By this it is able to capture the functionality that is offered by malleable signatures. It can be used for creating a blanket statement as legally introduced in Sec. 4.5.5 (see Sec. 18.7), but also to achieve the security goal of contingency (see Sec. 18.8 or publication n° 14 [379]). The notion of contingency, where explicitly any subsequent modification is authorized, shows that ACA requires to define what authorized means. This must be done by an integrity policy as identified in Analysis Result 8[331], i.e., also brought up by *Bishop* [51] (see Sec. 5.1.9). The policy is specified as part of the signature creation and checked during the verification process. It is out of this thesis' scope to discuss the technical issues of implementing policies for authorized changes further.

Note, ACA, as strange as it might sound, includes the ability that the authorization will be so broad that it grants any arbitrary change to any other party, e.g., everyone can be authorized to change everything[332].

If any modification is explicitly authorized to be done by anyone, then this is the dual to Integrity: Contingency. This is the 'grey area' between the clearly non offered integrity protection provided by UCA and the integrity protection offered by ACA (and of course NCA), which is depicted in Fig. 34. This grey area offers ACA, thus it still offers cryptographic protection. However, as it allows all changes it is no longer sensible to name this protection 'integrity protection'. In fact, this is the dual of integrity, a notion which has been termed contingency in [33] and was introduced by *Rost and Pfitzmann* [421]. This thesis discusses in more detail in Sec. 18.8 how the application of MSS (ACA–1CD–PUB) enables constructing a technical contingency protection mechanism.

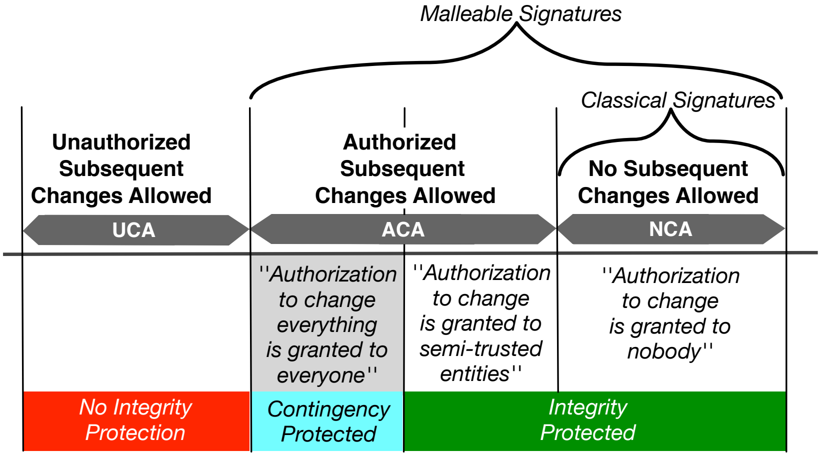

Malleable Signatures
Classical Signatures
Unauthorized Subsequent Changes Allowed
Authorized Subsequent Changes Allowed
No Subsequent Changes Allowed
UCA
ACA
NCA
"Authorization to change everything is granted to everyone"
"Authorization to change is granted to semi-trusted entities"
"Authorization to change is granted to nobody"
No Integrity Protection
Contingency Protected
Integrity Protected

Figure 34. Malleable signature schemes (MSS) and classical digital signature schemes (CDSS) regarding the first aspect of the extended integrity definition: Three levels (UCA, ACA, and NCA) of increasing strictness for tolerable subsequent modifications

6.3. 2^{nd} aspect: Detection of subsequent changes

The second aspect for differentiation is which modifications are detectable by a Verifier. The analysis of legal definitions for data integrity showed that the definition from ENISA does differ from the EU's definition for advanced electronic signatures in one respect: The Directive 1999/93/EC and Regulation 910/2014 allows integrity with subsequent modifications — regardless if they are authorized or not — and only requires that they are detected; ENISA explicitly forbids subsequent modifications (see Fig. 32). Thus, following ENISA no subsequent change is allowed and all occurred changes are required to be detected as unauthorized (NCA).

ENISA's interpretation is in line with the technical definitions of Stallings. While Gollmann's technical definition maps to the detection level of the EU signature legislation from Directive 1999/93/EC.

[331] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

[332] Usually the Sanitizer is limited in its subsequent modifications and can thus be semi-trusted (see Sec. 3.4.1 on page 47).

Different in their detection from the aforementioned are Clark and Wilson, TCSEC, and US HIPAA; they all allow authorized changes to take place, but lack a clear and explicit statement if these allowed changes can be performed undetected or must be detectable. That those definitions are interpreted in this thesis as implicitly allowing subsequent authorized changes (ACA) does not — yet — make a statement about the detection of the authorized changes. To overcome this, this thesis considers four levels of detection in two subgroups. This thesis will differentiate between two kinds of detections:

- **occurred change detection (*CD):** detection of *occurred* and materialised authorized modifications, and
- **potential future change detection (*FD):** detection of future *potential* authorized modifications to protected data.

The differentiator of the above two groups of detection is the time the modifications happens in relation to the time the detection happens: Detecting the possibility of future changes — the mere optional future possibility for a modification — is in one group (*FD) and the detection of already occurred change is in the other (*CD). The subgroups offer five for *CD and four levels for *FD; they are depicted in Fig. 35. This thesis found the given levels sufficient to separate and distinguish between existing schemes and definitions. However, the definition is not withstanding the possibility of being extended with more fine grained levels.

The focus of this thesis lies on the occurred change detection (*CD). Reasons are the legal Analysis Result 4[333] and the technical Analysis Result 10[334]; They both highlight that integrity protection requires to detect any occurred subsequent modifications. Additionally, this detection of an occurred modification is of relevance for the accountability: to transfer accountability away from the Signer also requires that the detectability of a modification that actually has happened is possible (see Analysis Result 12[335]). The interesting coupling of detection and accountability is further detailed in Sec. 6.4.7.

In the following, the levels of each group are briefly explained.

6.3.1. Occurred change detection (*CD)

The need to explicitly state an integrity protection mechanism's ability to detect occurred subsequent changes follows from the legal Analysis Result 4[333] requiring to detect any subsequent modification and the opposing cryptographic property of transparency (see Definition 157) of existing SSS and RSS.

- **no occurred change detection (NCD):** Any occurred modification to an integrity-protected message is unrecognisably hidden to a Verifier. This is the theoretical unique infimum; this allows ordering the levels in a lattice.
- **at least one unauthorized occurred change detection (UCD):** The Verifier detects that either no or at least one unauthorized modification to an integrity-protected message has occurred. The exact number of occurred unauthorized modifications or where they happened remains invisible to the Verifier.
- **at least one occurred change detection (1CD):** The Verifier detects that either no modification or that at least one modification (authorized or unauthorized) to an integrity-protected message has occurred. The exact number of occurred modifications or where they happened remains invisible to the Verifier.
- **all individual block occurred change detection (BCD):** For all blocks individually it is detectable for a Verifier if in that block either no or at least one modification occurred to that block's integrity-protected information.

[333] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

[334] Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification (see Sec. 5.5.3 on page 136).

[335] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

- **all change detection (ACD):** The Verifier detects that either no modification has occurred or if a modification occurred to an integrity-protected message, then all modifications are detectable by a Verifier offering the highest possible grade of detail. This is the theoretical supremum; this allows ordering the levels in a lattice.

Fig. 35 (a) shows them ordered in the resulting lattice.

NCD does not provide legally recognised integrity protection. Following the legal Analysis Result 4[a], occurred change detection of at least level NCA – 1CD or ACA – UCD is required for a legally recognised integrity protection mechanism. Analysis Result 4 states that a legally recognised electronic signature protects against undetected subsequent modifications, making no difference if they are authorized, malicious or accidental. Hence, this thesis would like to explicitly note that the theoretical unique infimum of the lattice, no occurred change detection (NCD), is not protecting integrity.

[a] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

UCD can be achieved if authorized subsequent modifications are not recognisable, but unauthorized modifications are detected. Note, in the case that all subsequent modifications are deemed unauthorized, i.e. for NCA integrity, the levels 1CD and UCD are non differentiable. This is in line with their meaning, detecting all modifications as unauthorized (NCA integrity) would require to detect all unauthorized modifications. Hence, NCA – UCD $=$ NCA – 1CD.
As motivated in Sec. 3.10.1 the property of cryptographic transparency for a scheme that authorizes subsequent modifications (ACA integrity) cloaks all but unauthorized modifications from the Verifier. If the Verifier could not on its own (ACA – $\geq$1CD – PUB) or with the help of other parties make the Sanitizer accountable for its modification, then a mechanism might still offer $\geq$ NCD protection. This detection is denoted as ACA – UCD. For ACA integrity mechanisms it is possible to order them as ACA – UCD $<$ ACA – 1CD.

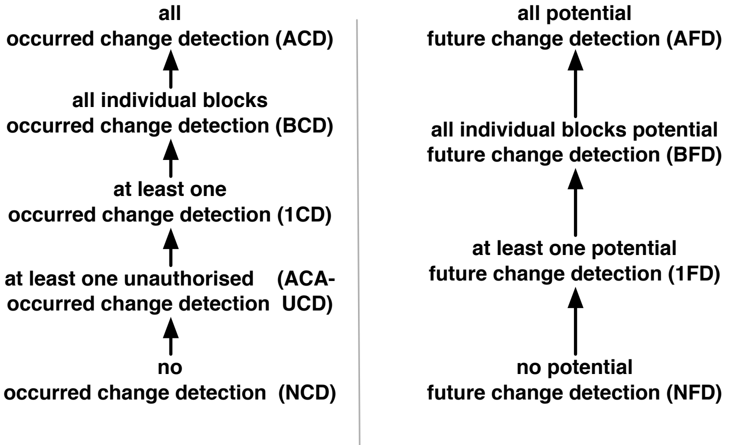
6.3.2. Potential future change detection (*FD)

The detection levels of *FD describe the level of detail with which a Verifier can identify what authorized modifications might happen in the future. In comparison the increasing *CD-levels are allowing a Verifier to identify in more detail what modifications had happened to integrity-protected messages in the past.

- **no potential future change detection (NFD):** The verifier cannot detect the potential of a future change to an integrity-protected message. This is the theoretical unique infimum. It allows ordering the levels in a lattice.
- **at least one potential future change detection (1FD):** The verifier detects that there is at least the potential that a subsequent change to the integrity-protected message could happen in the future. The exact number of potential changes or where they might happen remains invisible to a verifier.
- **all individual blocks potential future change detection (BFD):** For all blocks it is detectable for a Verifier if in that block at least one potential change to that block's integrity-protected information could happen in the future or if no future modification is possible. This means that the Verifier can find out which blocks are admissible for modifications and which blocks are not.
- **all potential future change detection (AFD):** All potential changes that could happen to an integrity-protected message in the future are detectable by a verifier offering a certain grade of detail, i.e. regarding their locations or types of changes. This is the theoretical unique supremum. It allows ordering the levels in a lattice.

The order is depicted in Fig. 35 (b).

The group of future change detection (*FD) allows to explicitly capture a property identified for SSS coined "strong transparency" [12]. However, the name is misleading and it has also not been formalised by *Ateniese et al.* in [12]. It is described to provide "[...] guarantees that the verifier does not know which parts of the message are immutable and thus does not know which parts of a signed message could potentially be sanitizable" [12]. This thesis would still call that NFD.


all
occurred change detection (ACD)
all individual blocks
occurred change detection (BCD)
at least one
occurred change detection (1CD)
at least one unauthorised occurred change detection (ACA-UCD)
no
occurred change detection (NCD)
all potential
future change detection (AFD)
all individual blocks potential
future change detection (BFD)
at least one potential
future change detection (1FD)
no potential
future change detection (NFD)

(a) occurred change detection (*CD)

(b) potential future change detection (*FD)

Figure 35. Integrity details: Several levels in two groups to differentiate the increasing level of details in the detection; (a) detection of occurred subsequent modifications (*CD); (b) detection of potential future subsequent modifications (*FD); Note: $\mathrm{ACA-UCD} < \mathrm{ACA-1CD}$ but $\mathrm{NCA-UCD} = \mathrm{NCA-1CD}$, hence it is denoted as $\mathrm{ACA-UCD}$

MSS can be NFD, but *FD-levels are not relevant for the discussion of the probative value, but for a generic notion of integrity. Even though the used cryptographic algorithms and the constructions for MSS are different to those of CDSS, the Verifier can only detect via the algorithm used for verification that a signature was generated using an MSS. Further to that, an MSS must neither leak the size nor the contents of the admissible blocks (ADM) to the adversary. Those MSS can then be classified as NFD.

During the legal analysis for an increased probative value the ability to detect that a document can, but was not subsequently modified, was not mentioned. This thesis found the *FD-levels to not impact the probative value nor the confidentiality. It is of interest to be able to cover the interesting property of invisible sanitizable parts proposed by *Ateniese et al.* [12] in 2005. Recently, the property proposed back then was formally defined and instantiated in joint work published by *Beck, Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [32] and by *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107] in 2017.

Knowing more details where future modifications can happen can also increase the knowledge where future modifications cannot happen. With more knowledge, i.e. a higher level of *FD, the Verifier will gain knowledge also about which block of the message are immutable. In this thesis this level is denoted BFD. In the extreme case, if a signature generated by an BFD-level MSS convinces the Verifier that all blocks are not (or no longer[a]) admissible for authorized subsequent modifications, then the Verifier is able to convince itself that no more subsequent modifications will happen in the future.

[a] If the scheme allows to remove existing authorizations this property was termed "consecutive sanitization control" [347] by *Miyazaki et al.*.

6.3.3. Levels from *CD and *FD can be combined in one scheme

Theorem 1 : *CD and *FD are combinable

*A level in the *CD group can be combined with a level in *FD group. Especially, they are not all mutually exclusive.*

Proof 1

Theorem 1 is proven by providing a concrete example for a combination: There exists a scheme that allows the Verifier *to detect that a modification has occurred (1CD) while still allowing the scheme to hide the detection of future potential changes (NFD). NFD is what Ateniese, Chou, de Medeiros, and Tsudik termed "strong transparency" [12]. The construction is described based on an* SSS *for multiple sanitizers as follows: "[...] declare* *every block of the message mutable, but assign public keys of non-existing (dummy) censors [in this thesis called* Sanitizer*] to the blocks the signer wishes to remain unmodified." [12]. Using this idea as a recipe to build upon a multi-sanitizer* SSS *which offers BCD – PUB (like [69]) yields a scheme where the* Verifier *could not know if a message can be sanitized in future, or if the secret sanitizing key corresponding to the stated public key was just a dummy. However, due to the BCD – PUB for each block the* Verifier *on its own can detect the change, which yields an* SSS *that offers ACA – BCD – NFD – PUB integrity.* □

Note, the above is possible as the levels are describing aspects of integrity seen from the point of the Verifier. Examples for further combinations are schemes that offer transparency (UCD) and which still allow to identify potentially admissible blocks ($\geq$ NFD), e.g. [12], or schemes that are UCD and only 1FD, e.g. [107].

Achieving certain levels of *FD is interesting in their own right, but out of the thesis' scope. The analysis of legal texts indicated that *FD is not important for the judgement of the probative value (see Sec. 6.3.2), hence it is not further discussed in this thesis. The capability called *FD was introduced in the initial SSS paper by *Ateniese et al.* [12]. However, it has not been formalised in [12]. Only recently a formalisation and a construction has been given by *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107] (see Appendix A publication n^{o} 55).

Further discussion of the levels in *FD is skipped in the remainder of this thesis. Henceforth, it is not further declared what level of future change detection (*FD) a scheme offers. The reason being that no analysis yielded that *FD was important for the judgement of the probative value (see Sec. 6.3.2).

It is not part of this thesis to further research to identify if levels of *CD and *FD are independent. Maybe not all combinations of levels of *CD and *FD can be constructed.

6.4. 3rd aspect: Determine 3rd-party verifiability and accountability for subsequent modifications of messages

Accountability is legally important following Analysis Result 5[336] to identify the legal signatory and following Analysis Result 7[337] to make sure that the signatory can be assumed to understand if any subsequent modifications were authorized by him. Technical accountability is required to use it as technical evidence to solve legal disputes. To assess the probative value and legal value of MSS the question how accountability is determined in MSS and if that is different to classical DSS becomes important.

Following Analysis Result 12[338], accountability allows a rightly complaining participating party to initiate a dispute resolution protocol that outputs which party is accountable. The analysis of accountability and non-repudiation notions carried out in this thesis (see Sec. 5.2) shows that accountability requires that at least for all rightly complaints the accused protocol participant cannot construct a proof that convinces a third party of him not being accountable. In this line, this thesis sees accountability as a property of a protocol which allows that when "a protocol participant (rightly) complains that something went wrong, then it should be possible to (rightly) hold specific protocol participants accountable for their misbehavior"[339] [302]. In this spirit, this thesis defines the property of accountability for modifications that were done subsequent to the application of integrity protection as follows:

Definition 108 : Accountability for subsequent modifications of messages

An integrity protection method satisfies the property of ***accountability****, if and only if a protocol participant for an integrity-protected message, for which the integrity verification algorithm outputs that the integrity is valid, can construct a proof that*

a) either corroborates the absence of subsequent modifications,

[336] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[337] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[338] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

[339] The American English spelling from original work is retained for terms and quotes.

b) or corroborates that a subsequent modification has happened, and establishes at least one of the below:

b_1) the correct and active involvement of an authorized participant other than the Signer[340]*,*

b_2) the absence of an authorization from the Signer *for the occurred subsequent modification (s),*

such that the proof convinces a third party.

To become accountable for a modification it is required to detect that modification. If the party judging the accountable party cannot detect any subsequent modification, then there is no reason other than to corroborate the absence of subsequent modifications[a]. However, for any subsequent modification that is detected, the above notion is in line with legal Analysis Result 5 and allows that a rightly complaining participating party can initiate a dispute resolution protocol that outputs which party is accountable. A discussion on this follows in Sec. 6.4.7.

[a] and this again falls into case *a* from Definition 108

The accountability defined in Definition 108 is general with respect to the involved parties in the generation of the non-repudiation evidence. In the notion of accountability given in Definition 108 it is not mentioned — on purpose to stay general — what is required to construct the proof. Thus, it is general enough to cover cases where the execution of the dispute resolution protocol might require additional knowledge, e.g. secret keys, or participation of other protocol participants.

The general accountability from Definition 108 explicitly incorporates a requirement for 3^{rd}-party verifiability of the accountability. For accountability to be usable in practice and to use the integrity protection mechanism and its accountability features to convince third parties, the integrity mechanism must offer 3^{rd}-party verifiability.

This was defined as follows:

Definition 24 : (Unrestricted) 3^{rd}-party verifiability

> *3^{rd}-party verifiability guarantees that on receipt of a verifying evidence the verifying party can easily and publicly disseminate the evidence and convince an unlimited number of third-party verifiers about the truth of the evidence. The above must work without the verifying party's need to disclose any secrets of the initial verifier or generator. Third-party in the above refers to any party other than the creator of the evidence or the initial verifier of the evidence.*

Indeed, different to classical legally accepted digital signature schemes (CDSS) the previously existing malleable signature protocols for integrity require the involvement of other parties, namely the Signer, to achieve the construction of a proof of accountability that can be presented to a third party. Two forms of how integrity protection mechanisms allow to determine the accountability will be differentiated in this thesis. The differentiation between the two levels is based on the need of the dispute resolution protocol to involve protocol participants or their secrets or trusted parties during the correct construction of a proof of accountability for modifications. The two levels of 3^{rd}-party verifiable accountability are:

- non-public correctly-executed and non-repudiable interactive (INT), and
- non-interactive public (PUB).

Additional levels of accountability can be constructed to describe mechanisms that give rise to designated verifiers as initially introduced by *Jakobsson et al.* [269] as they can be combined with authorized subsequent modifications as presented by *Derler et al.* [146] for RSS or from using the "undeniable" [292] aspect of SSS raised by *Krawczyk and Rabin*.

[340] Signer here means the participant that applied the integrity protection. Note, ISO 10181-6 denotes this party as the 'initiator' [340].

Classic DSS offer non-interactive public accountability (PUB) assuming prior key distribution. Accountability of a classic DSS gives non-interactive public accountability (PUB): The Verifier runs the Verify algorithm on its own with the Signer's public verification key. This allows the Verifier to assure itself that the algorithm is correctly executed. On an output of `valid` the Verifier knows that it will be able to present the message, the signature and the Signer's public key as a proof of the absence of modifications to third parties. The Verify algorithm will reproduce the exact same output of `valid` for any any other party correctly executing the Verify algorithm and also trusting that the Signer's public key is related to the Signer.

Note, Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) is assumed as stated in Sec. 3.8.3. Thus, any interactions with third parties usually involved in PKI, like checking the revocation status of the public key certificate, or obtaining it in the first place, are assumed to be excluded and henceforth do not count as an interaction.

In the domain of MSS it would be legally favourable that "[i]n case of a dispute the signatory must be able to prove that certain modifications have been done by a certain redactor" [460], if subsequent modifications have been done by a dedicated Sanitizer. The above statement was made by *Stranacher, Krnjic, Zwattendorfer, and Zefferer*. It is one of the authors' requirements for the use of RSS in eGovernment [460]; *Stranacher et al.* make this observation based on — and citing — published results of this thesis[341]. This is in line with the technical definitions of accountability found in MSS schemes (see Sec. 5.5.5 for an analysis and Analysis Result 12[342] for the resulting requirement [343]).

However, the above given notion by *Stranacher et al.* still does not require that this attribution can be made without a trusted arbitrator; Following *Stranacher et al.* accountability can be interactive non-public (INT) **or** non-interactive public (PUB). This is in line with the analysis of this thesis, as even though their requirement, which contains a strong "must" [460], can be fulfilled with correctly-executed and non-repudiable interactive non-public accountability (INT). *Stranacher et al.* [460] do not discuss this in detail. In the remainder of this section, e.g. Subsections 6.4.1 and 6.4.2, the drawbacks of the non-public interactive accountability (INT) is discussed. In relation to the underlying assumptions of correctness and benign interaction this thesis finds that non-interactive public accountability (PUB) is not only increasing the probative value but also the general legal value of signatures generated by MSS.

6.4.1. Interactive non-public (correctly-executed and non-repudiable) accountability (INT) for integrity-protected messages

The general accountability property, as defined in Definition 108, requires that a protocol participant can construct the proof to convince a third party of the accountability of a specific protocol participant. In the case that a Verifier in the MSS wants to prove the accountability of the Signer or Sanitizer the general notion of accountability allows for parties other than the Verifier to be required to establish accountability. This is captured as Definition 109. The correctly-executed and non-repudiable interactive accountability given in Definition 110 captures the required involvement of a participant with access to the secret keys or by an additional trusted third party that makes the dispute resolution protocol non-repudiable.

Obtaining trustworthy public keys is assumed and not counted as an interaction. Following Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) any interactions with third parties to obtain trustworthy public keys henceforth do not count as an interaction.

Definition 109 : Interactive accountability for integrity-protected messages (INT)

An integrity protection method satisfies the property of ***interactive accountability****, if and only if a protocol participant for an integrity-protected message, for which the integrity verification algorithm outputs that the integrity is valid, can construct a proof that*

a) ***either*** *corroborates the absence of subsequent modifications,*

[341] In [460] *Stranacher, Krnjic, Zwattendorfer, and Zefferer* cite the joint publications n° 1 and n° 11 listed in Appendix A.

[342] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

[343] The requirement to proof that a redaction has taken place is obviously also in line with the results of this thesis: Requirement 4 formulates the technical requirement of accountability in Sec. 7.4.

*b) **or** corroborates that a subsequent modifications has happened, and establishes **at least one** of the below:*

b_1) the correct and active involvement of an authorized participant other than the Signer[344]*,*

b_2) the absence of an authorization from the Signer *for the occurred subsequent modification (s),*

such that this proof convinces a third party[345],
*but the construction of that proof **requires at least one** of the below:*

- *the active involvement of another protocol participant, or*
- *the active involvement of a trusted third party, or*
- *the knowledge of another participant's secret keys.*

Note, interactive and non-public accountability (INT) incorporates a requirement for 3^{rd}-party verifiability of the integrity.

Further, Definition 109 does not explicitly capture that for accountability of the level INT it is assumed that the interaction is always successful and the results are always correctly computed. The latter is fine to assume with cryptographic algorithms and is captured as correctness properties (see for example Definition 147). However, to enforce correct execution might be necessary if it is beneficial for parties not to cooperate. To enforce such compliant interaction or at least record and attribute non-compliant behaviour of the interacting party might require the involvement of a trusted third party (TTP), i.e., make the dispute resolution protocol non-repudiable by using a TTP. The level of involvement and the level of trust required for the TTP might differ as described in [512]. For the following it is enough to assume that it can be enforced. One way to achieve it is by entrusting the trusted third party with the required secrets and by having the TTP run the required algorithms on behalf of the Signer (or Sanitizer). However, the TTP then becomes more than functionally trusted (see Definition 21). In the case of signature generation keys this imposes additional problems to not violate the legal requirement to keep the secret signature generation key under control in a QSCD or SSCD as stated in Analysis Result 6. For the remainder it is out of scope how it can be achieved, but that it requires additional overhead should be noted. The adjusted accountability definition that makes this explicit is as follows:

Definition 110 : Interactive non-public correctly-executed and non-repudiable accountability for integrity-protected messages (INT)

*An integrity protection method satisfies the property of **interactive accountability**, if and only if a protocol participant for an integrity-protected message for which the integrity verification algorithm outputs that the integrity is valid can construct a proof that*

*a) **either** corroborates the absence of subsequent modifications,*

*b) **or** corroborates that a subsequent modifications has happened, and establishes **at least one** of the below:*

b_1) the correct and active involvement of an authorized participant other than the Signer[346]*,*

b_2) the absence of an authorization from the Signer *for the occurred subsequent modification (s),*

*such that this proof is **non-repudiable by the accused party**[347] and convinces a third party,*
*but the construction of that proof **requires at least one** of the below:*

- *the active involvement of another protocol participant, or*
- *the active involvement of a trusted third party, or*
- *the knowledge of another participant's secret keys.*

[344] Signer here means the participant that applied the integrity protection

[345] So far this is equal to Definition 108.

[346] Signer here means the participant that applied the integrity protection.

[347] This is the explicit statement different to Definition 109.

6.4.2. Impact on accountability and legal certainty when a benign and correctly-executed and non-repudiable interaction with the Signer is required

In a nutshell, while the cryptographic algorithms found in MSS assume to always yield the correct and same response, e.g. all parties always interact and act according to the protocols, in real-life the enforceability might depend on the party requesting the interaction. In the harmonised MSS algorithms presented in Sections 11.3 and 11.10 there are two algorithms involved for interactive accountability (INT): Judge and Proof. The latter algorithm Proof allows the Signer, if and only if the Signer is not accountable, to produce a so-called proof π for a valid message-signature pair with which the former algorithm Judge identifies the Sanitizer as the originator. If no subsequent modification took place, the Signer is not able to calculate such a proof and thus Proof will output the Signer as accountable.

EXAMPLE

Consider the following example interactions: Assume Alice wants to know if Bob is accountable for a certain message m^+ for which Bob's signature, that was generated with an MSS, verifies. Assume that Bob is not accountable for that message m^+ and the corresponding signature, because it was generated by a valid authorized subsequent modification. Alice wants to establish hard evidence, i.e., Alice wants to get evidence which is verifiable and reproducible later. Alice plans to be able use the evidence to base her legal case upon it, thus the evidence shall convince a court. Using an MSS, the two algorithms involved in generating the evidence for accountability are Judge and Proof. Assuming correct interaction with the Signer in the dispute resolution protocol, Bob will always supply the asking party with the correct output π generated by the Proof algorithm. To recall, a correctly working Judge algorithm on input of the correctly computed output of the Proof algorithm will cryptographically correctly accuse the right party, i.e., Signer or Sanitizer. The above intertwining and need of correct interactions is the reason to term it *correctly-executed and non-repudiable* interactive accountability (INT).

However, the real interaction is beyond the scope of the integrity protection mechanisms and thus it would be possible for Bob to supply a wrong answer to Alice on purpose, or even to provide no answer at all. In this example it is assumed that even though Bob would be able to prove correctly that the message m^+ and the corresponding signature is not from him, he does not do so. On each interaction of the proof resolution protocol Bob can freely decide how to react. Bob thus provides a wrong answer π^* in the dispute resolution protocol when it is initiated by Alice. Here, the assumption is that only the output of MSS's Proof algorithm being returned in the protocol. Then, in case of a wrong answer, Alice, based on the algorithms of the MSS, will establish that Bob is accountable for m^+. The problem occurs when Alice later involves a third party called Dread — maybe the judge in her court case. Bob now repudiates that he is responsible. Note, Bob's answer (or non answer) to Alice was assumed to be the bare proof and nothing else. π^* back then was failing to accuse the Sanitizer, however it is non-repudiable towards a third party. Alice could have just manipulated Bob's answer to falsely accuse him. Hence, the third party will need to re-involve Bob in another run of the dispute resolution protocol. Maybe Bob is told not to lie because he is 'under oath'. Unfortunately Alice could only 'assume' that Bob acts in the same way as previously in her interaction with him. Instead, Bob now correctly participates in the dispute resolution protocol initiated by Dread and produces technical evidence to cryptographically repudiate being accountable for m^+ by sending a correct π. Dread, as well as Alice and any one else, can now use Judge and π to find out that a Sanitizer is accountable and not Bob. Alice's fails to use the fact that the answer π^* that made her Judge algorithm accuse Bob as a proof of Bob's accountability towards a third party. Because just the π^* following the definitions of the algorithms of RSS and SSS on its own does not provide a proof that it came from Bob or that it relates to m^+ or the signature on it.

The above example illustrates that a Verifier that engages with the Signer in a scheme that offers interactive accountability (INT) must additionally ensure that the protocol is correctly-executed and non-repudiable and can not be technically repudiated. This additional requirement on the properties of the dispute resolution or disavowal[348] protocol is outside the scope of the RSS and SSS algorithms described in this thesis.

Assuming correctly-executed and non-repudiable non-public interactive accountability (INT) achieves 3^{rd}-party-verifiability as defined in Definition 24 but with uncertainty for accountability of the Signer. If not correctly-executed and non-repudiable, the Verifier might assume wrongly that the Signer is accountable for the verifying message. Note, in all existing and new MSS schemes

[348] A term from undeniable signatures [372].

that offer the cryptographic property of Signer- and Sanitizer-accountability, a **correctly** generated proof by the Signer which allows him to technically repudiate the signature is transferrable to third parties. However, the assumption of a correct execution must be made when only non-public interactive accountability (INT) is achieved. The reason is that a Verifier can not check if a proof π is a result of a correct execution of Proof unless Judge technically accuses Sanitizer. Therefore, in cases where Judge accuses the Signer for the provided π the Verifier has *legal uncertainty* for MSS which offer only public interactive accountability (INT).

To even better understand uncertainty, this thesis interprets the opposite notion: *legal certainty*. Legal certainty in the light of a technical signature mechanism is defined to have the following meaning:

Definition 111 : Legal Certainty for Electronic Signatures

> *Legal certainty means that there exists sufficient certainty in advance that a court will assign a high probative value to a specific electronically signed document, due to the circumstance that the respective technical signature mechanism was used.*

Contrary to the INT notion of accountability, a classical DSS allows to offer what this thesis terms *non-interactive public accountability* (PUB). Solving the problem of legal uncertainty being introduced by interactive accountability (INT) requires non-repudiation to be added to the interaction, e.g., by a third party. However, no interaction is preferred[349]: For a digital signature scheme, there is already the assumption that there is an off-line functionally-trusted third party[350] which is needed to initiate the initial trust into public keys. This is obligatory in order to achieve the linking of the signatory, i.e., the natural person, to the technical signature verification key as stated in Analysis Result 5[351]. This is built upon to offer *non-interactive public accountability* (PUB) as defined next in Sec. 6.4.3.

6.4.3. Non-interactive public accountability for integrity-protected message (PUB)

Non-interactive public accountability allows a protocol participant to construct the proof to convince a third party of the accountability of a misbehaving specific protocol participant without the active involvement of another protocol participant; and without the active involvement of a trusted third party; and without the knowledge of additional secrets. The only needed knowledge is the knowledge needed for integrity verification, i.e., the other participant's public keys or his own secret keys. Definition 112 captures that any Verifier with a trusted public key of the Signer and the Sanitizer can correctly decide whether a valid message-signature pair originates from the Signer or from the Sanitizer without interacting with the Signer or Sanitizer or a third party.

Definition 112 : Non-interactive public non-repudiable accountability for subsequent modifications of messages (PUB)

> *An integrity protection method satisfies the property of* ***non-interactive non-repudiable public accountability****, if and only if a protocol participant for an integrity-protected message for which the integrity verification algorithm outputs that the integrity is valid can construct a proof that*
>
> *a)* ***either*** *corroborates the absence of subsequent modifications,*
>
> *b)* ***or*** *corroborates that at least one subsequent modification has happened, and establishes* ***at least one*** *of the below:*
>
> *b_1) the correct and active involvement of an authorized participant other than the Signer[352],*
>
> *b_2) the absence of an authorization from the Signerfor the occurred subsequent modification (s),*

[349] Also following the discussion of [512].

[350] See Sec. 2.10.1 for the definition of notions regarding TTPs.

[351] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[352] Signer here means the participant that applied the integrity protection

such that this proof is ***non-repudiable by the accused party*** *and convinces a third party*[353]*,* *but the construction of that proof does* ***not require any one*** *of the below:*

- *the active involvement of another protocol participant, or*
- *the active involvement of a trusted third party, or*
- *the knowledge of another participant's secret keys.*

Henceforth, the notion of 'non-interactive public accountability' or 'PUB' is used to refer to non-interactive non-repudiable public accountability (PUB) and it incorporates the requirement for 3rd-party verifiability of the integrity.

6.4.4. Relation between non-interactive public accountability and interactive accountability

A protection mechanism that provides non-interactive public accountability (PUB) must not require a correct interaction with the Signer. A protection mechanism that provides non-public interactive accountability (INT) must require a correct interaction with the Signer. These two forms, even in their non-repudiable forms, are contrary to each other but are not ordered.

The two forms (PUB and INT) of the accountability aspect do not form a lattice. While not stating that it is impossible, this thesis did not try to order the two required forms of how integrity protection mechanisms allow to determine the accountability in a lattice. Introducing a level of no accountability as the theoretical infimum is possible for the aspect of accountability. However, ordering the two forms of PUB and INT as well is out of scope of this thesis. It is left for future work to find a suitable infimum and supremum (see Sec. 19.3).

Each form corresponds to existing cryptographic definitions. In particular INT corresponds to a cryptographic definition of properties found in existing MSS termed Signer- and Sanitizer-accountability. The form of PUB corresponds to the cryptographic properties found in existing CDSS and will allow to increase the legal certainty (see the follow-up discussion in Sec. 6.4.5). Note, that the notion of PUB also achieves the combination of two properties introduced by *Pfitzmann* [372]. Namely, "[s]trong requirement of the recipient on disputes" [372][a] in combination with "[s]trong requirement of the signer on disputes" [372][b].

[a] "Strong requirement of the recipient on disputes. The recipient wishes that any message he has accepted should also be accepted by a third party in a subsequent dispute." [372].

[b] "Strong requirement of the signer on disputes. [...] In this case, one need not find the signer to carry out a fair dispute. " [372].

6.4.5. Legal certainty increases with non-interactive public accountability

If one assumes that no errors prohibit the successful execution of the protocol, and that a participating party can provide evidence to refute the accusation of accountability, then the failure of any party to participate in a dispute resolution protocol can make that party become appointed accountable for the message. This correctness assumption requires additional enforcement, in order to prohibit a party in practice from choosing to behave incorrectly in selected executions on purpose. A malicious party can incorrectly re-act in one protocol run, while behaving correctly at a later time in another dispute resolution protocol run for the same message. In the failed-on-purpose protocol run the failing party is accused, due to its failure to participate or due to its failure to provide refuting evidence, while successfully refuting the accusation in the second protocol run. This impedes creating legal certainty (as introduced in Definition 111 on page 153) for the Verifier, and only allows certainty for Signers. Thus, interactive accountability for integrity-protected messages (INT) creates a disequilibrium between Signer and Verifier.

[353] So far this is equal to Definition 110.

This problem is circumvented by having non-interactive public accountability: no interaction is required[354]. In terms of the SSS, interactive accountability for integrity-protected messages (INT) works on the assumption that the Judge algorithm repeatedly gets input from the Proof algorithm and both are always correctly executed. Accountability being public and non-interactive (PUB) removes the dependence of the Proof algorithm on input generated from non-public values by Proof. In a scheme that offers PUB, each successful verification with just publicly available information, i.e. with the public signature verification key(s), generates the same strong evidence that a successful and correct interaction would have created. As such, the non-interactive public accountability evidence generated can be re-validated by third parties on their own, as it requires no interaction or additional knowledge. Moreover, the public version overcomes availability problems. Hence, non-interactive public accountability gives legal certainty[355] and removes the burden of an on-line (see Sec. 2.10.1 for a definition) involvement of the trusted party and forgoes the need to run interactive dispute resolution protocols.

6.4.6. Focus lies on 3^{rd}-party verifiability of integrity and accountability and thus on transferable signatures

In general, integrity could further be differentiated into:

internally verifiable: The verifiable state of data integrity is verifiable only by the party that has applied the integrity protection mechanism to the data.

3^{rd}-party verifiable: The verifiable state of data integrity is verifiable by all third parties just by giving them access to the public keys required for verification.

The notion 3^{rd}-party verifiable is in line with the notion of public verifiability as defined in Definition 100. Public or 3^{rd}-party verifiability could not be achieved with symmetric keys, i.e., by a symmetric message authentication code (MAC) [278][356]. The aspect of being internally verifiable is not used further, because this thesis aims to increase the probative value by generating evidence which is valuable to settle disputes in front of a third party during the protection of integrity. To be used as evidence that convinces a third party, the Verifier actually needs the signature to be transferable, as defined in Definition 101. Hence, this secondary aspect is not used to further differentiate the accountability aspect of integrity in this thesis and is left as future work (see Sec. 19.3). For generality the proposed extended integrity notion left this open, which makes it generally applicable also to symmetric mechanisms like MAC. Contrary, for the focus of this thesis, 3^{rd}-party verifiability is always needed and hence incorporated into the accountability notion (see Sec. 6.4) by stating that "this proof convinces a third party" in Definition 110 for INT and Definition 112 for PUB. Further, the aspect of allowing the accountability to be used to provide non-repudiation services as defined in Definition 88 on page 121 is explicitly stated in the accountability definitions of correctly-executed and non-repudiable INT (Definition 110) and PUB (Definition 112).

6.4.7. Level of occurred change detection (*CD) and scope of accountability are related to each other

The level of detectability of occurred changes influences the scope (see also Sec. 6.5) of accountability for occurred modifications. In the following three cases, this thesis always presumes that the signature still verifies:

NCD This is a theoretical infimum and offers no integrity protection, hence no accountability. Thus, it is not further considered.

[354] No interaction because Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) as stated in Sec. 3.8.3 on page 58 means interactions to require public keys, like obtaining keys or checking the validity of qualified public key certificates, are assumed to have been executed already and thus do not count as interactions in this context of the evidence generation step.

[355] See Definition 111 on page 153 for the term.

[356] A MAC based on a symmetric key forbids an adversary to simply re-calculate the integrity protection check value, as the value can only be computed correctly with the knowledge of the secret key. However, it does not offer the same level of authentication of origin, nor does it allow to prove the integrity to third parties. The reason is simple: While the shared secret is needed for verification, who ever possesses it can also generate an integrity check value for any modified message.

UCD When the Verifier can not (neither by itself nor by help of a Arbitrator) verify that there was a modification by a Sanitizer, i.e. as in NCD, then the Signer will always be accountable for the whole message even if all or only a small part of it was modified. Thus accountability remains with the Signer unless an unauthorized modification happens.

1CD When the Verifier (by itself or by help of a Arbitrator) is not able to identify the position of a modification to a block, but can identify that there was a modification by a Sanitizer, i.e., as in 1CD, then the Signer will not be accountable for the whole message even if only a small part of it was changed. In the same manner, a modification in one block of the signed message will make the Verifier (by itself or by help of a Arbitrator) attribute the accountability for the whole message to the Sanitizer. With a detectability level of 1CD the Sanitizer, by running the Sanit algorithm, takes over the accountability of the complete message (see also the notion of 'touching' a message as described in Sec. 3.4.3).

≥1CD When a Verifier (by itself or by help of a Arbitrator) is able to detect that an authorized modification has taken place on a block of the message, i.e., as in BCD or ACD, then the accountability of the Sanitizer can be limited to that block. The Signer can be seen as being accountable for the unchanged and the non-admissible parts, as well as for authorising the occurred subsequent modifications.

Hence, having block-level occurred change detectability (BCD), an increased level of detail, will also increase the detail of accountability. Vice versa, decreasing the level of detail on detectability will also decrease the detail of accountability.

6.5. Scope of block-level and message-level

In the joint publication [391] published at ARES the property of transparency was identified to be defined on message scope as well as on blocks (see Appendix A publication n^{o} 6). This section extends this and introduces block-level and message-level properties as a general concept for integrity properties.

Some integrity aspects will differ in respect to whether the information obtained through the integrity protection verification concerns only the message as a whole or is more detailed and generates information about individual block-level properties. When a property, like detectability allows to make statements of the 'whole message only' then this property is said to have the scope of the message, or in short a *message-level* property. When a property has the scope of the block this thesis identifies them as *block-level* properties.

EXAMPLE

For example, the BCD level of the detectability aspect means that the integrity protection offers detectability as a block-level property.

6.6. Proposed extended integrity definition

Following is this thesis's proposed extended definition of Integrity; a discussion follows afterwards.

Definition 113 : Data integrity

Data integrity is a specific state of data that is verifiable (potentially by a third party).
A verification of data integrity that yields a ***valid*** *output corroborates that the integrity-protected data has* ***not*** *been modified (or destroyed) in an unauthorized manner* ***since***
the integrity protection mechanism has been applied to the data.
A verification of data integrity that yields an ***invalid*** *outcome corroborates that the integrity-protected data has been modified (or destroyed) in an unauthorized manner* ***after***
the integrity protection mechanism has been applied to the data.

Notes for Definition 113:

Note 1 *Definition 113 does not exclude authorized modifications.*

Note 2 *Definition 113 requires to precisely capture in an integrity policy (e.g., ISO 10181-6 [340]) what constitutes an unauthorized or authorized change for this data. Compliance with this integrity policy is implicitly assumed to be checked to the full extent by the integrity verification mechanism.*

Note 3 *Definition 113 sees data integrity in accordance with ISO 10181-6 as "a specific invariant on data" [340]. Following this, the integrity protection mechanism "detects the violation of internal consistency" [340]. "A datum is internally consistent if and only if all modifications of this item satisfy the relevant integrity security policies." [340]*

Note 4 *Definition 113 is not concerned with the external consistency (e.g., as in [125]), or veracity [215] of the data.*

Note 5 *Definition 113 is hinting that integrity should allow the detection of integrity violations by third parties. However, it is generic in this respect and thus the level of detection (as defined in Sec. 6.3) may vary, and it depends on the technical detection achieved by the mechanism.*

Note 6 *Definition 113 explicitly leaves it open to define if the current state of integrity can be checked by a third party or not with an additional property.*

Concerning the research question regarding the probative value of an MSS, it is important to see integrity as a state describing data which is *verifiably* "[...] as it was supposed to be [...]" [214] since the integrity protection was applied. Hence, the Definition 113 explicitly expresses that integrity is a state that data can have. Thus the thesis integrity definition follows: TCSEC [308], *Gollmann* [214], Regulation (EC) No 460/2004 [179], US HIPAA [484, clause 164.304].

Further, the integrity definition spells out that it can be verified. Due to its generality the integrity notions leaves to define by which entities it can be verified. Note 6 of Definition 113 can be seen as redundant, but it clarifies this again explicitly. As Definition 113's first line "[...] verifiable (potentially by a third party)" hints, the thesis' goal to define requirements for an increased probative value requires the integrity notion to be 3^{rd}-party verifiable (see Sec. 6.4.6). As a result, integrity should not only be verifiable by the party that has applied the integrity protection mechanism to the data, but must also be verifiable by other third parties, which also goes beyond the party which is receiving the integrity-protected data in a transmission[357].

Further, Definition 113 is in accordance with [33, 214, 421]: It acknowledges that data could be "supposed" to undergo changes and that the data's integrity status shall remain unchanged for those changes. This option to make room for authorized modifications (ACA) is explicitly indicated in Note 1.

Definition 113's Note 2 clarifies that the integrity protection mechanism must codify the application's policy for subsequent modifications. This means that the application of an integrity mechanism fixes several aspects: (1) whether or not it tolerates subsequent modifications to signed and thus integrity-protected data, (2) which subsequent modifications are authorized and (3) which subsequent modifications are detectable by the Verifier. In this the integrity definition of this thesis follows ISO 10181-6 which states that integrity of an "item" is maintained if "[...] all modifications of this item satisfy the relevant integrity security policies [...]" [340].

As stated in Note 4 and Note 5, Definition 113 does not consider the protection of wrong information to be a problem of data integrity, following *Gollmann* [215].

Note 6 of Definition 113 leaves the possibility to determine the level of verifiability depending on the application. As discussed in Sec. 6.4.6 this can range from unrestricted 3^{rd}-party verifiability, over designated verifiability, to only internal verifiability. If the mechanism is 3^{rd}-party-verifiable the verification procedure's output can serve as a cryptographically-strong evidence towards a third party. This allows the integrity mechanisms to become the basis for accountability. As noted, this is of particular interest in this thesis to increase the probative value. However, the proposed definition of integrity is generic enough to cover also integrity protection mechanisms for applications where internal integrity verification would suffice. Note, designated verifiability is out of scope as discussed in Sec. 2.10.2.3.

[357] Put into more networking oriented words: Message-level rather than transport-level integrity is preferred. However, the proposed integrity notion is still general enough to allow it.

6.7. Extended integrity notion represented by the triple of the three aspects

This thesis describes the achieved or required integrity protection using a combination of levels along all three aspects. Henceforth, integrity protection is stated as a triple:

$$\text{protection}_{\text{integrity}} = \begin{array}{l} (\ \text{Level of Granted authorized Modifications,} \\ \ \ \ \text{Level of Detectability of Modifications,} \\ \ \ \ \text{Form of How to Determine Accountability for Modifications}\ \ ) \end{array}$$

This added precision allows to capture more details than existing integrity definitions including the differences of existing malleable signatures as shown in Sec. 6.8. The shorthand notation just hyphenates the abbreviations, e.g. NCA–1CD–PUB integrity. This shorthand is used in the remainder of this thesis. Further, this thesis found the need to describe several levels for aspects: a minimal level[358] or the exclusion of a certain level. Henceforth, statements regarding the levels — including talking about all levels — will be denoted by the symbols *, $\geq$ and ! as follows:

- * **All levels**: If no explicit level is stated in the existing definition, this is denoted by a star, e.g., *CD indicates that any level of detection is suffice, this includes even the minimal level with missing detection of occurred modifications (NCD).
- $\geq$ **Minimal level**: When a minimal level is needed, then this gets denoted by $\geq$, e.g., $\geq$1CD means any level including and greater than[359] 1CD detection is suitable.
- ! **Exclusion of a level**: To negate a level it is marked with an exclamation mark, e.g., ! NCD means any level offering a form of detection that is not NCD is suitable.

Note, equivalence of levels, i.e. ! NCD has the same meaning as $\geq$UCD, might only be due to the specific way the levels are defined in this thesis. If there was an intermediate level between NCD and UCD, then this level would still be below $\geq$UCD while being ! NCD.

6.8. Existing technical integrity definitions in extended integrity notation using a triple of three aspects

Tab. 3 showcases how the new extended notion is generally applicable to describe existing notions of integrity and how it allows to capture the subtle differences that were found during this thesis' analysis. Fig. 32 already offered a visual overview of existing definitions along two — of the three — aspects. Analysis Result 11 showed differences in the integrity notion, an overview was given in Tab. 1.

Due to NCA having the possibility of being emulated by a certain policy in ACA, all existing definitions that allowed NCA and ACA (see Fig. 32) are shown as ACA in Tab. 3. This thesis found only the definition by *Biba*[360] [47] to state explicitly its form of accountability, i.e., *Biba* allows the non-public interactive form of 3^{rd}-party-verifiable accountability (INT). When the analysis found that an integrity definition tolerates authorized subsequent modifications it is marked as ! UCA; only if the definition contains an explicit statement, the decision between ACA or NCA could be made and got noted in Tab. 3 on page 159.

[358] Possible if the levels are ordered. Of course also $<$ and $>$ would be possible relations on the ordered levels.

[359] The greater-than relation follows the lattice from Fig. 35.

[360] See Sec. 5.1.1 on page 110.

Definition	**shorthand of additional aspects**
Integrity Considerations of Secure Computer Systems by *Biba*	ACA – ≥1CD – INT [361]
Data Integrity by *TCSEC*	! UCA – ≥1CD [362]
Data Integrity by *Clark and Wilson*	ACA [363]
Integrity by *ISO 2700x*	! UCA [364]
Data Integrity by ITU X.800	! UCA – ≥1CD [365]
Data Integrity by *ISO 7498-2*	! UCA – ≥1CD [366]
Data Integrity from RFC 4949	! UCA – ≥1CD [367]
Integrity of Network Messages by *Stallings*	NCA – ≥1CD [368]
Data Integrity by *Gollmann*	! UCA – ≥1CD [369]
Data Integrity by *Bishop*	ACA – ≥1CD [370]
Digital Signature by *ISO 7498-2*	NCA – ≥1CD [371]
Digital Signatures by ITU X.800	NCA – ≥1CD [372]
Integrity framework by ISO 10181-6	! UCA – ≥1CD [373]
EF-CMA secure Digital Signatures by *Goldwasser et al.*	NCA – ≥1CD [374]
Digital Signatures by *NIST FIPS 186-4*	! UCA – ≥1CD [375]

Table 3. Existing integrity-related computer science definitions restated with the new aspects

6.9. Visualisation of extended integrity notion

This thesis devised a visual representation to allow for a more intuitive understanding of the levels. It facilitates a visual comparison for classic as well as different malleable signatures discussed in the literature and the remainder of this thesis. It indicates the level of protection offered including the above three aspects, with one small exception: On the aspect of detectability the visualisation of future change detection (*FD) capabilities is omitted and only the detectability of occurred changes (*CD) is visualised. Further, it is visualised if the protection is on the scope of block or on the whole message (see also Sec. 6.5). An overview of the visual representation's four quadrants is shown in Fig. 36.

In the centre the visualisation shows if the integrity protection method has the vital property for allowing to 'detect that at least one unauthorized modification has occurred in the message'. This is in the central position — the bull's eye — as without this property there would be no integrity protection.

[361] "objects may not be directly modified by subject possessing insufficient privilege" [47, p. 30] and interpreting "only proper modifications are made" [47, p. 14] as the detection to be in need to detect all non-authorized modifications but allow transparency (INT).

[362] Data must "not [have] been exposed to accidental or malicious alteration" [308].

[363] While this is a preventative integrity notion, meaning a system that offers data integrity according to *Clark and Wilson* has to prevent any integrity violation from happening, it explicitly allows "well-formed transactions" [125].

[364] Data must retain "accuracy" [249]; this leans towards the notion of 'veracity'.

[365] Data must "not [have] been altered or destroyed in an unauthorized manner." [263] and "detects any modification" [263].

[366] Data must "not [have] been altered [...] in an unauthorized manner" [239] and "[...] detects any modification, insertion, deletion or replay of data [...]" [239, p. 5].

[367] "unauthorized or accidental manner" [438] and "ensuring that changes to data are detectable" [438].

[368] "messages are received as sent" [452] but also mentions that protection is concerned with detection.

[369] "as it was supposed to be" [214] gives ! UCA and "[...] integrity check functions provide the means to **detect** whether a document has been changed;" [214] is ≥1CD.

[370] "a security policy identifies authorized ways in which information may be altered and entities authorized to alter it" [51, p. 97] but sees a need for detection to "[...] report that the data' integrity is no longer trustworthy." [51, p. 5].

[371] NCA as "signature can only be produced using the signer's private information" [239], and ≥1CD as the protection is against "forgeries"' [239], without specifying what a forgery is.

[372] NCA as "signature can only be produced using the signer's private information" [263] and "detects any modification" [263] for connection or selective field integrity which results in its detection being ≥1CD.

[373] "checks integrity-protected data for modifications" [340].

[374] NCA because only the Signer can generate valid signatures as "[...] the adversary needs to generate a fresh message that verifies under a previously fixed public verification key with non-negligible probability. A message is fresh if the adversary never queried the signing oracle for this message." [211]; detection is described in the verification algorithm that outputs "[...] true if and only if [the signature] is valid.".

[375] "detect unauthorized modifications" [355] and "detect whether or not the information was modified after it was signed" [355].

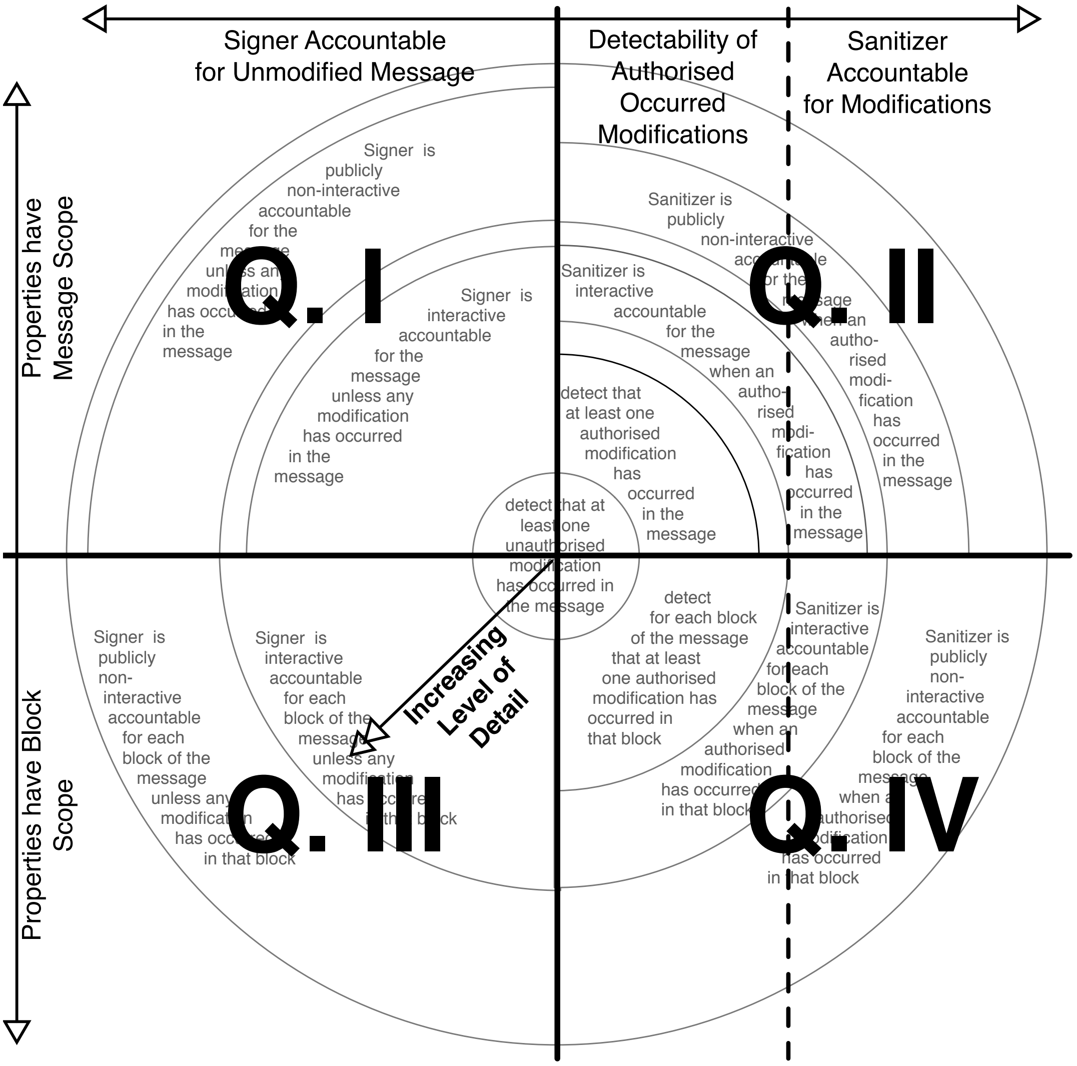

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Q. I
Q. II
Q. III
Q. IV
Increasing Level of Detail
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
detect that at least one authorised modification has occurred in the message
Sanitizer is interactive accountable for the message when an authorised modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an authorised modification has occurred in the message
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block

Figure 36. Four quadrants; on the x-axis the levels of Detectability and thus Accountability; on the y-axis message-level or block-level in general; to the outside an increased level of detail from the complete message to blocks

In the upper half (Q.I and Q.II) the properties have 1CD detection. They have a scope of message-level for the properties.

In the lower half (Q.III and Q.IV) the properties have an increased detection of BCD. BCD allows detecting the property of integrity for each block individually, for all blocks of the message. This gives the offered properties a scope of block-level.

In the left half (Q.I and Q.III) the properties indicate that the integrity protection method to have accountability of the Signer till a modification of the message is detected. Note, these properties can be supported regardless of the mechanism offering NCA or ACA.

In the right half (Q.II and Q.IV) this thesis has grouped properties that only make sense if authorized modifications are allowed, e.g., if the integrity protection is offering ACA. When authorized changes (ACA) are granted the different properties visualised in the right half are differentiating how the mechanism attributes accountability of the Sanitizer once an authorized modification has happened.

Being further away from the centre indicates that accountability is moving from the interactive accountability (INT) form to the non-interactive public accountability (PUB) form.

6.10. Application of the newly proposed definition: Classic digital signature schemes (CDSS) in the extended integrity definition and visualisation

To show the applicability of the extended integrity notion and the usefulness of the visualisation this section classifies a classic digital signature scheme (CDSS) using the extended integrity definition showing the generality of the newly refined integrity and detectability notions. Later, this thesis will use the new extended integrity classification to describe the properties of existing (in Chapter 11) and later newly designed malleable signature schemes (in Chapter 13 and 14).

As an example for a CDSS this thesis selected the RSASSA-PSS from PKCS-v2.2 [422]. If this scheme is instantiated using suitable algorithms with legally accepted strength, it is a legally accepted technical building block sufficient to generate electronic signatures with a high probative value. In RSASSA-PSS from PKCS-v2.2 [422] the signature process in a nutshell is as follows: First, pad the bit-representation of the data to a length suitable for the following algorithms; second, compute a hash-value of the padded data using a cryptographically secure hash-function (i.e., SHA-256); third, with an asymmetric encryption algorithm (i.e., RSA) compute the signature for the hash-value. As a result of the secure hash-function's properties, a change of one bit in the signed bit-representation results in a different hash-value and invalidates the signature.

Thus, classic digital signatures protect the bit-representation against any subsequent modification. Hence, all subsequent changes are unauthorized (NCA) and a single change is detected (1CD). Due to the construction of the hash-functions it cannot be differentiated if the input differed in 1 bit or more than 1 bit. Hence, only 1CD can be achieved in the detection. Moreover, a subsequent modification is detected and a non-repudiation evidence verifiable by third parties (PUB), including the Verifier, is provided. The evidence makes the Signer accountable as long as no subsequent modification has happened. A subsequent modification will result in a failed verification outcome and thus the Signer will be no longer accountable. Generating this evidence requires no further interactions (PUB) because all interactions regarding obtaining trustworthy public keys are assumed to have been executed already[376].

Hence, the classic digital signature with a secure cryptographic hash offers **NCA–1CD–PUB integrity** protection. In the visualisation, the properties that related to schemes that do not allow authorized modifications all are grouped in quadrant one (Q.1), i.e., in the upper left corner. Fig. 37 shows the visualisation for properties offered by a classic digital signature (DSS) like RSASSA-PSS from PKCS-v2.2 [422] always assuming the correct use of suitable algorithms with a sophisticated key lengths.

[376] Follows from Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed).

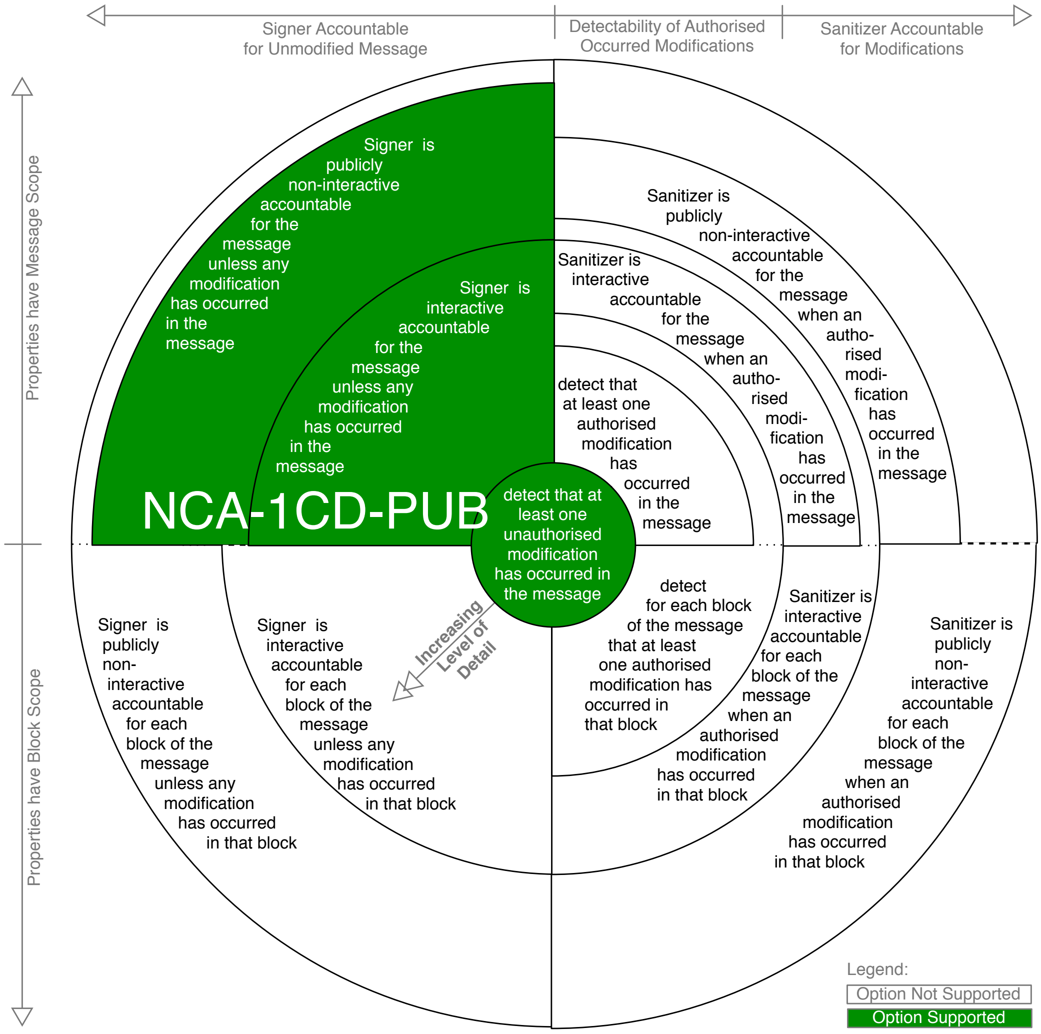

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
NCA-1CD-PUB
detect that at least one unauthorised modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
Increasing Level of Detail
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 37. Classical digital signature schemes (CDSS) based on a secure cryptographic hash-function, RSASSA-PSS from PKCS-v2.2 [422], offer NCA–1CD–PUB integrity, because Verifiers detect any subsequent modification without distinguishing between authorized or unauthorized modification and verification proves the Signer's accountability for unmodified messages non-interactively, i.e. by using just the Signer's public verification key

7 Technical requirements for integrity and authenticity protection to gain a high probative value

Overview of Chapter 7

This chapter gives legally adequate and rigorous technical descriptions of security functionality, especially regarding the security goals of integrity and authenticity, that technical mechanisms must achieve to increase the probative value of electronic documents and to earn the legal statutory evidentiary presumption awarded by the EU electronic signature laws, i.e., Regulation 910/2014 or the older Directive 1999/93/EC. To recall, statutory evidentiary presumption would make the electronic signature "legally equivalent to hand-written signatures" [Regulation 910/2014] and then EU member state laws would award the signed data a high probative value. Henceforth, these requirements will be used in Sec. 17.3 to evaluate if the newly designed mechanisms and cryptographic properties presented in this thesis allow to award statutory evidentiary presumption to a signed electronic document protected with an MSS.

Chapter 7 uses the extended integrity notion given in Chapter 6 when referring to integrity. This follows from the methodology as the integrity terminology was required to be refined before being able to give the technically precise requirements[377]. The key words MUST, MUST NOT, SHOULD, SHOULD NOT, MAY, and MAY NOT indicate the need to fulfil a functionality as described in RFC 2119 [62]. Henceforth, this thesis uses the word MUST (or MUST NOT) to indicate a mandatory property that is needed in order to offer an increase in the probative value.

SHOULD is used to indicate a strong recommendation given by this thesis in order to streamline the legal argumentation for an increased probative value technically easier. Requirements are marked as SHOULD and not as MUST because the legal analysis did not identify them as explicitly mandatory. However, compliance with strong recommendations (SHOULD) is aimed at in this thesis as it allows for a clearer line of argumentation that those mechanisms legally increase the probative value. Requirements indicated with MAY are optional and allow signatures to be additionally used for certain applications.

Chapter 7 concentrates on digital signatures as a mechanism for the "detection of integrity compromise" [340] — following the analysis obtained in Sec. 5.5.6 and the focus of the thesis as discussed in Sec. 1.4.1.

[377] This chapter is related to step 2.2 (Formalise application requirements (including legal) using technically precise terminology) of the methodology from Sec. 1.5.

7.1. Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer)

Legal Analysis Result 4[378] stated that Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4). It further requires producing integrity protection such that the protection is "linked to the data [...] in such a way that any subsequent change in the data is detectable" [Art. 26 d Regulation 910/2014][Art. 2 2.d Directive 1999/93/EC]. Only then the legal regulations directly award the document with a high probative value.

To fulfil any signature related functionality it is required that the signatory can define what the data is that needs the integrity protection, following Analysis Result 9[379] and the signature functions as stated in Sec. 4.5.1. Further, Analysis Result 8[380] stated that applications define what constitutes a modification and that a policy to describe such a modification can be assumed (see Sec. 5.5.1), which yields the following: The need for the integrity protection is related to — or defined and controlled by — the application and within often closely aligned with the structure present in the data to be protected.

The usual approach to cater for the above is as follows: The integrity protection mechanisms' implementations will take care to protect relevant information contained or encoded within the data's structure. For MSS this can be achieved by a decomposition (see Definition 38) into blocks which closely follows the structure, e.g., each element in a set that is to be protected for integrity is a block (see Definition 35).

To identify in more detail what is the needed integrity protection this thesis follows *Liu, Lu, and Yip*; they divided the goal of integrity into three sub-goals: *content integrity*, *structural integrity* and *context referential integrity* [318]. Note, the original definition from [318] is given in the context of XML documents, which are ordered trees. Thus, the original definitions were tailored to this data structure. In this respect, this thesis generalises them and defines content, structural and referential integrity protection as general concepts for arbitrary data structures.

This results in the following requirement:

Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer)

An integrity protection mechanism for a blocks MUST *allow to protect the integrity of content (see Definition 114), structural integrity (see Definition 115) and referential integrity (see Definition 116) of the blocks within the integrity-protected message as requested and defined by the party who applies the mechanism.*

Notes for Requirement 1:

Note 1 For Requirement 1 all integrity protection sub-goals are defined at the level of blocks. All blocks together form the data that is to be integrity-protected, i.e., all blocks form the message or document[a]*.*

Note 2 Following Requirement 1, the integrity protection of the message and subsequent modifications to this message can include content, structural and referential integrity protection or modifications to any of the three, i.e. in any mixture.

[a] The terms 'message' and 'document' are used synonymously here.

Note 1 and 2 explicitly clarify that for each block it is defined what is required to be protected, i.e., *content integrity*, *structural integrity* and *referential integrity*. Message-level properties can be constructed by requiring the block-level property to hold for all block(s) of the message. Note 2 updates explicitly a central notion of this thesis: Henceforth, the notion of 'modified message' incorporates modifications not only to content, but also to structure and references. Thus, the reader needs to consider this when taking into account already stated definitions, e.g., the non-interactive public accountability for subsequent modifications of messages (see Definition 112).

[378] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

[379] Analysis Result 9: Definition of authorized and unauthorized modification; and modification in general (see Sec. 5.5.2 on page 133).

[380] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

7.1.1. Requirement 1.1 (Mechanism MUST protect content integrity as requested by the Signer)

Liu et al. define content integrity for XML messages as follows:

> "The XML data content refers to element name, attribute, and values of an element or sub XML data. Content integrity means that XML data content will not be changed or destroyed in transmitting or storage. This is ensured by generating a hash-value value of XML data." [318].

Content integrity in relation to a document composed from block(s) then captures the integrity protection of the actual contents of a single block, e.g., before being signed the block's content is uniquely reversible serialised into a string of bits.

Definition 114 : Content integrity protection

> *A mechanism offers **content integrity protection** for a block $m[a]$, if $m[a]$'s contents are integrity-protected.*

7.1.2. Requirement 1.2 (Mechanism MUST protect structural integrity as requested by the Signer)

The structure of blocks — alongside content — can carry information that shall be protected. The existing description for structural integrity was given by *Liu et al.* for XML elements and reads as follows:

> "An XML data structure integrity protects the location information of an element in XML data. It means that if the location of an element in the XML data has been changed, it will lead to an invalid verification. Location information of an XML element refers to the position of this element in the XML data. Element location information consists of three parts: parent, level, and order in sibling." [318].

This thesis further assumes — following the original notion by *Liu et al.* which discusses this per XML element — that the protection of the complete document's structure is not required. Structural integrity protection is for a single block within a structured message composed of blocks. Hence, this thesis assumes this property to be at the level of blocks.

EXAMPLE

Example: As a start assume that content integrity is protected for a tree-structured document. Further assume the protection mechanism is meant to protect an ordered tree. However, assume it fails detecting a changed order of siblings; instead it protects all parent-child relationships between the blocks and each block's height in the tree only. Then this mechanism does not protect structural integrity of any of the blocks in the ordered tree. Important is that the Signer controls — and consents to — the structural integrity protection. If the Signer decides to consent to protect's scope for the same tree-structured document to protect the blocks structured as an **un**ordered tree, then the previously mentioned protection mechanism would indeed successfully protect all structural information *selected* to be protected. Hence, the same protection mechanism would fulfil the protection requirements of structural integrity.

This thesis defines structural integrity in relation to a structured document made from blocks as follows:

Definition 115 : Structural integrity protection

> *A mechanism offers **structural integrity protection** for a structured document's block $m[a]$, if **all selected** structural properties for the block $m[a]$, which are uniquely identifiable from the structured document or the integrity protection checksum (e.g., signature) by the Verifier, are integrity-protected.*

Only structural properties that the Verifier can extract from the document and its integrity protection value could become structurally integrity-protected. Definition 115 assumes that the protected structural properties can be uniquely recovered from the structured document and its integrity protection. This allows the Verifier to identify which structural information contained in a document are integrity-protected. It does not require the Verifier to be able to identify all those that were initially protected by the Signer. Hence, this thesis's definition allows to withhold initially protected structural information from verification by a certain Verifier, i.e., structure might be hidden (e.g., by stenography [279, 389]) from one Verifier, but still be protected and verifiable by another Verifier. This makes the definitions compatible with designated verifier signature schemes [313, 456, 497].

7.1.3. Requirement 1.3 (Mechanism MUST protect referential integrity as requested by the Signer)

Relations between two or more blocks' contents can carry information worth protecting the integrity of. Furthermore, relations between between blocks even without their contents could be defined. The existing explanation given by *Liu et al.* termed this context referential integrity, the original explanation was given for an XML element is as follows:

> "In other words, an XML element has an entire meaning only when related to other elements in the same XML data, and these elements have been defined as context- related elements in this paper. Context referential integrity is used to protect context-related elements of an element in XML data. It will provide a binding between an element and context-related elements. This means if context-related elements of an element have been altered, it will also lead to an invalid verification." [318].

Transferred to blocks, the above means that rather than just protecting a single block $m[a]$ individually towards its content and its structural position the protection scope for the information stored in $m[a]$ must be wider as it is adversarially affected by a modification of $m[b]$ from the same document. Context referential integrity assumes that the integrity of a single block $m[a]$ is violated — even if the block itself is intact and in place, i.e., content and structural integrity hold for $m[a]$ — because of an integrity violation that occurred in the contextually related block $m[b]$.

Hence, this thesis defines referential integrity in a structured document made from blocks as follows:

Definition 116 : Referential integrity protection

Notation: The document has ℓ block(s) denoted as $m[1], \ldots, m[\ell]$. Block $m[a]$ has relation(s) to block(s) $m[b_1], \ldots, m[b_i]$, where $1 \leq a \leq \ell$, $1 \leq b_i \leq \ell$.

*A mechanism offers **referential integrity protection** for a block $m[a]$ in relation to block(s) $m[b_1], \ldots, m[b_i]$ if the integrity of $m[a]$ is valid only if for **all related** block(s) $m[b_1] \ldots m[b_i]$ the integrity (content, structural or referential as individually selected) is also valid.*

Notes for Definition 116:

Note 1 Definition 116 does not define how relations are captured, only that they are defined during integrity protection generation and checked during referential integrity verification.

Note 2 The relations in Definition 116 could introduce cycles for the verification process. Hence, Definition 116 assumes that those cycles are detected and correctly handled by the verification algorithm.

7.2. Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier)

Following Analysis Result 2[381] a reduced integrity protection that authorizes subsequent modifications (ACA) is legally acceptable within Regulation 910/2014 and Directive 1999/93/EC. Additionally, any subsequent modification must remain detectable, which is also legally required if the integrity protection which authorizes no subsequent modifications (NCA) is legally not enough as found in Analysis

[381] Analysis Result 2: Legal acts do not explicitly forbid authorized modifications (see Sec. 4.6.2 on page 101).

Result 4[382]. Further, Analysis Result 11[383] shows that ACA is also described in some existing technical integrity notions. In order to hold the signatory accountable and thus to uphold legal Analysis Result 5[384], the technical mechanism MUST allow to verify that the specific level of integrity protection offered for a protected document was indeed applied with the consent of the signatory. Because only with this consent the potential subsequent modifications can be proclaimed as authorized by the Signer, as highlighted in Analysis Result 7[385]. Especially, this consent seems necessary to capture the definition of scope and protection level with which the electronic signature will achieve the conclusory functionality (see Sec. 4.5.1): Conclusory functionality means here that the signatory is finally selecting the scope which is protected with the selected strength by the generated signature. In exchange for this consent of the Signer — and hence to conform to Analysis Result 7[385] (see Sec. 4.6.7) — the document gains legal accountability of the signatory even in case of an authorized subsequent modification. This includes the need to know and define a policy as found in Analysis Result 8[386]. Already in the previous Requirement 1 paved the way by encoding all of the above — consent, modification definition, and choice and control of needed protection — into the statement that integrity protection must be 'as requested and defined' by the Signer. In this regard, the following Requirement 2 makes this now explicit:

Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier)

An integrity protection mechanism MUST *enable the Signer to*

1. *be able to control and make an informed decision about the scope and strength of the integrity protection,* ***and***
2. *present a proof of consent for potential subsequent authorized modifications arising from the Signer's decision, such that it is verifiable by a third party.*

This mandates a proof of consent by the Signer for the chosen scope and strength of the integrity protection. The proof MUST *be verifiable by a third party[a]. However, it does not need to be verifiable by the Verifier or the third party alone, nor without involving other participants, i.e., it could be interactive[b]. To minimise problems[c], the proof of consent* SHOULD *be non-interactively and publicly verifiable[d] by the Verifier and by a third party.*

Further, an integrity protection mechanism for a blocks SHOULD *enable the Signer to*

3. *present a proof of consent such that it is non-interactively and publicly verifiable by the Verifier and also by a third party, which means that it is verifiable without*
    - *the active involvement of another protocol participant, or*
    - *the active involvement of a trusted third party, or*
    - *the knowledge of another participant's secret keys.*

[a] Taking also the role of Verifier, but being an additional and different party.

[b] Comparable to the aspect of accountability denoted as INT (see Sec. 6.4.1).

[c] See Sec. 6.4.2 for a discussion of the negative impact that the interaction requirement has on the aspect of accountability.

[d] Comparable to the aspect of accountability denoted as PUB (see Sec. 6.4.3).

Requirement 2 is comprised of two obligatory and one recommended function:

- First, the control of scope and strength of the digital signature mechanism MUST be exercised and consented by the Signer for the integrity — split in content integrity (1.1), structural (1.2) and referential integrity (1.3) — for **each protected** block. This is discussed in more detail as Requirement 2.1.
- Second, the proof of consent MUST to be verifiable by third parties, such that it can be used as evidence in court. This is discussed in more detail as Requirement 2.2.

[382] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

[383] Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect (see Sec. 5.5.4).

[384] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[385] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[386] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

- Third, it recommends that the proof of consent SHOULD be verified non-interactively and publicly by the Verifier and also by a third party. This is discussed in more detail as Requirement 2.3.

Finally, note that the conclusory function from the list of functionalities postulated by BT-Drs 14-4987 [147] requests that the signature by its location[387] shall protect the whole document (see Sec. 4.5.1 for the list). Namely, BT-Drs 14-4987 states for electronic signatures that the hash-value gets calculated on the whole document and can only be calculated once the document is fully conceived[388]. However, as discussed in Sec. 4.5.5, an underspecified document, i.e., a document with still missing content, can be legally created: This is legally known by the term *blanket statements*. When signing the blanket statement the signature cannot capture all the contents of the declaration of intent, as some contents are still missing at the time of signature. Instead it — additionally to the contents at the time of signature — captures the signatories intention to leave it underspecified. Following the German § 172 BGB this can be achieved by a letter of authorisation, that gives authority to the delegatee to subsequently fill the gaps left by the underspecification. Note, this letter of authorization might be required to be in written form.[389] Following *Boemke and Ulrici* the conclusory function is not regarding time[390], but regarding location [55, p.191–192]. On paper, still following the arguments given by *Boemke and Ulrici*, consent for the carried out subsequent later modifications can be provided by a signature made on the bottom of a blank paper and by that knowing that content can get later inserted above it [55, p. 192][391]. Then the signature below the content on paper can in principle still provide the conclusory function.

7.2.1. Requirement 2.1 (Mechanism MUST enable the Signer to control scope and strength of the integrity and authenticity protection)

The Signer must be able to set the scope and the strength of the integrity and authenticity protection. This allows to achieve Analysis Result 7. Block(s), which are not signed, are not under the scope of the integrity protection; and block(s), which are signed, are integrity-protected that offers a defined strength regarding the block(s)'s integrity and authenticity in all three integrity sub-goals (i.e., content, structural, and referential integrity).

This thesis termed this property Signer control[392] and defines it as follows:

Definition 117 : Signer Control

> *A malleable signature scheme offers* ***Signer control****, if the Signer, and only the Signer, can determine, which data is permitted to be subsequently altered. in which manner.*

For the Signer to be able to exercise sufficient control the integrity and authenticity protection mechanism MUST specify and thereby fix **all** of the following:

1. which blocks of an integrity-protected message are possibly modifiable by fixing the integrity policy (thereby fixing UCA, ACA, NCA; as defined in Sec. 6.2);
2. which modifications are authorized (redaction, sanitization; as defined in Definition 105);
3. the level of detection of modifications and specify if an occurred modification can be detected by Verifiers (*CD-level; as defined in Sec. 6.3.1);
4. the level of accountability (PUB/INT; as defined in Sec. 6.4).

[387] In German "der räumliche Abschluss eines Textes" [BT-Drs 14-4987].

[388] In German "Der Vorgang des elektronischen Signierens bezieht sich auf das gesamte Dokument, da zunächst aus dem Gesamttext ein Hashwert gebildet wird, der mit dem privaten Signaturschlüssel signiert wird. Die auf das Dokument bezogene Signatur kann daher erst nach der Erstellung des Textes gebildet werden. Diese logische Verbindung zwischen signiertem Text und Signatur stellt somit die Abschlussfunktion sicher." [BT-Drs 14-4987].

[389] "[...] die Ermächtigung zur Ausfüllung eines Blanketts [bedarf] selbst der Schriftform." [55, p.192]; source given by [55] states "Bürgschaftsblankett § 766 BGB, Schriftformerfordernis erstreckt sich (entgegen § 167 Abs. 2 BGB) auch auf die Bevollmächtigung zur Erteilung einer Bürgschaft;" ruling by German Federal High Court of Justice from 29.02.1996 [79].

[390] "Da der Rechtsverkehr bei einer vollständigen Urkunde nicht ergründen kann, ob zuerst der Text oder die Unterschrift auf dem Schriftstück vorhanden waren, gilt die Vorgabe des aus Gründen des Vertrauensschutzes nicht in zeitlicher Hinsicht." [55, p.191–192].

[391] "Eine formwirksame Erklärung ist selbst dann abgegeben, wenn der Erklärende zunächst ein leeres, aber unterschriebenes Blatt willentlich aus der Hand gibt ("Blankounterschrift") und ein von ihm Ermächtigter das Blatt nachträglich mit dem Erklärungstext versieht." [55, p.191–192].

[392] Apart from the term 'control' this is not comparable to a property termed "consecutive sanitization control" [347] by *Miyazaki et al.*.

Signer control is limited in existing RSS. This thesis opted to make it optional for the property of Signer control that the Signer can specify the entity (or entities) that is (are) authorized to perform such modifications, i.e., appoint the Sanitizer(s). In existing RSS the Signer has no control over the entities that can be in the role of the Sanitizer — redaction is a public redaction operation. See also Sec. 3.7 on page 56 for the differences in the authorization capabilities between CDSS, RSS and SSS. However, controlling which block(s) are authorized to be redacted is still a form of control.

Note, this thesis proposes a solution $pubacc\mathcal{RSS}$ (see Sections 14.13 and 14.14) that offers accountability and increased control over the entities that can redact.

Note, it is not required to fix the level of *FD, i.e., specify if the potential for modifications can be detected by Verifiers (see Sec. 6.3.2).

Note, the scope and strength of the mechanism must be fixed, i.e., all three levels *CA, *CD-level, and PUB/INT. This can be either fixed by the actual design of the chosen technical mechanism or fixed by a specific integrity policy to define the scope and strength of a chosen technical mechanism (see Analysis Result 8[393] for policies and Analysis Result 9[394] for defining authorizations).

7.2.2. Requirement 2.2 (Mechanism MUST verifiably document Signer's consent for chosen integrity and authenticity protection)

In order to hold the signatory accountable, following legal Analysis Result 5 and Analysis Result 7 it must be verifiable that the certain level of integrity protection that is offered, and the subsequent changes it authorized, was applied with the consent of the signatory. As per Requirement 2.1 it is possible for the Signer to use a certain mechanism with a fixed scope and strength or set a certain integrity policy for a mechanism, to define the scope and strength. This choice of integrity protection mechanism and integrity policy is interpreted as informed consent. This results in Requirement 2.2 which additionally to the required functionality of Requirement 2.1 listed as items 1. to 5. mandates the following:

The integrity and authenticity protection mechanism MUST

1.–5. same as in Requirement 2.1

6. produce a statement of consent of the Signer that it was the Signer's intention to apply exactly this protection as defined in above 1–5 of Requirement 2.1 to the document ;
7. produce that statement of consent such that it is verifiable by third-parties;

Note, the latter still allows that parties other than the Verifier need to be engaged to verify it. This is enough to fulfil the requirement following from Analysis Result 7. The following optional Requirement 2.3 makes the verification of consent easier for the Verifier.

7.2.3. Requirement 2.3 (Mechanism SHOULD enable Verifier to verify the proof of consent non-interactively and publicly)

To minimise any negative impact that a possible interaction might have[395] the proof of consent SHOULD be non-interactively and publicly verifiable by the Verifier and by another third party. This requirement is explicitly only for the consent to a potentially reduced scope and strength of the integrity protection; in essence it offers comparable functionality for the proof of consent that the non-interactive public accountability (PUB) demands for accountability, which was as an aspect of accountability (discussed in Sec. 6.4.3).

Additionally to the details specified as 1.–7. in Requirement 2.1 and Requirement 2.2 the integrity and authenticity protection mechanism SHOULD allow the Signer to:

1.–7. same as in Requirement 2.2

[393] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

[394] Analysis Result 9: Definition of authorized and unauthorized modification; and modification in general (see Sec. 5.5.2 on page 133).

[395] Compare the discussion on Sec. 6.4.2 on the aspect of accountability.

8. produce that statement of consent such that it is **non-interactively and publicly** verifiable **by the Verifier and also** by third-parties, which means that it is verifiable without
   - the active involvement of another protocol participant, or
   - the active involvement of a trusted third party, or
   - the knowledge of another participant's secret keys.

This requirement is in line with the property of public verifiability and transferability of the signature (see Definition 101 for definition and Sec. 5.3.7 for further discussion). Note, existing legally recognised CDSS also achieve public verifiability and they are transferable.

7.3. Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification)

For the verification of integrity the legal Analysis Result 4[396] demands that the signature must be linked to the data signed therewith in such a way that any subsequent change in the data is detectable. Moreover, electronic signatures have to fulfil the *legal verification function* (in German 'Verifikationsfunktion') among other functionalities prescribed in German legal texts [147, 149] (see Sec. 4.5.1). The verification function is described in [147] to enable

- a Verifier of a qualified electronic signature to check in a **simple way**
- if the document was **created using the private signature generation key** — assumed to be controlled by the Signer — , and
- if the document has been subsequently modified.[397]

Legally the above verification function is closely related to the identity function (in German 'Identitätsfunktion') and the authenticity function (in German 'Echtheitsfunktion'). Further, the archiving or integrity function (in German 'Perpetuierungsfunktion' or 'Integritätsfunktion') must be fulfilled. In order to achieve this Requirement 3 in combination with Requirement 4 has been formulated. Requirement 3 is as follows:

Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification)

The mechanism for integrity protection MUST *allow the verification of integrity and authentication of origin by the Verifier, such that it*

1. *enables the Verifier to identify the absence of any unauthorized subsequent modifications,* ***and***
2. *enables the Verifier to identify, only by the help of a known public key and without active involvement of any party that additionally knows the Signer's or Sanitizer's secret keys, that the corresponding secret key has been involved and as such the entity holding the secret key is the origin of the protected document.*

To enable an easy verification the mechanisms SHOULD *allow to identify the absence of any unauthorized subsequent modifications by the Verifier*

3. *without active involvement of any party that additionally knows the Signer's or Sanitizer's secret keys.*

Note, the verification of integrity and authentication of origin can be achieved also if it requires the help of other parties (INT). However, the legally accepted CDSS are non-interactively and publicly verifiable.[398] In its general form Requirement 3 requires that verification of integrity (Requirement 3.1) and must be achieved in either of the two forms INT or PUB. However, this thesis additionally strongly

[396] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

[397] In German the signature law interpretation in [147] was required an electronic signature to allow the following: "[...] kann der Adressat in **einfacher Weise** überprüfen, ob die Signatur **mit dem privaten Signaturschlüssel des Schlüssel-Inhabers erstellt** worden ist und ob der signierte Text **nachträglich verändert worden** ist." [147]; bold face has been added for highlighting.

[398] See Sec. 6.10 for an analysis of RSASSA-PSS from PKCS-v2.2 [422] being classified as NCA–1CD–PUB

recommends that both SHOULD be achieved in a non-interactive public fashion (PUB). Hence, for this operation it is marked as a strong recommendation that the operations SHOULD be non-interactive (Requirement 3.3). For authentication of origin (Requirement 3.2) it is argued that INT is not enough and that it MUST be non-interactive.

7.3.1. Requirement 3.1 (Mechanism MUST verify the absence of any subsequent modifications, i.e., ≥1CD integrity)

Analysis Result 9[399] and Analysis Result 10[400] state that the data's integrity MUST be protected such that authorized and unauthorized modifications can be defined (following from Analysis Result 8[401]) and detected. Having any subsequent modification detected, i.e., offering ≥1CD integrity, the protection mechanism fulfils the legal archiving or integrity function[402] and also the legal verification function described in German legislative texts [147, 149] (see Sec. 4.5.1). Note again that [147, 149] itself are not German law but an official interpretation stating the rational behind the former German signature law. It is not enough to offer ≥UCD protection as it fails to reliably detect any subsequent modification because only authorized modifications must be detected with UCD, while authorized might go undetected.[403] ≥1CD is required to fulfil Requirement 4[404]. The public form is highly recommended because the mechanism then exhibits the same behaviour towards the Verifier as current CDSSs (see Sec. 6.10 for an analysis of RSASSA-PSS from PKCS-v2.2 [422] being classified as NCA–1CD–PUB). See Requirement 3.3 in Sec. 7.3.3 for the recommendation to achieve the non-interactive public form, i.e., ≥1CD–PUB integrity.

7.3.2. Requirement 3.2 (Mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory)

Following Analysis Result 5 and to fulfil the identity function (in German 'Identitätsfunktion') of [147, 149][405] the origin of messages MUST be authenticated. Authentication is part of the cryptographic functionality of digital signature algorithms and it establishes that the corresponding secret key $\mathsf{sk}_{\mathrm{sig}}$ or $\mathsf{sk}_{\mathrm{san}}$ was used to generate the signature, which was verified as `valid` when providing the corresponding public key(s) to the verification algorithm. However, the functionality needed to fulfil this signature functionality goes — far — beyond that. It requires establishing the connection between the used secret key and an involved party that is represented by this key pair. If a signature scheme based on asymmetric cryptography (see Definition 13) is used, this connection can be achieved by using public key infrastructures. Hence, this requirement includes the explicit request for 'asymmetric cryptography to allow the origin's public key to be linked to a legal signatory'. Technically this is also defined as strong authentication, which provides authentication of origin and non-repudiation. To establish the link the public key used to verify the signature, which was generated by a secure signature scheme providing unforgeability and non-repudiation, must additionally be associated with exactly one legal signatory. The details to accomplish this, e.g., by a PKI, remain out of scope (see Sec. 1.4.3).

[399] Analysis Result 9: Definition of authorized and unauthorized modification; and modification in general (see Sec. 5.5.2 on page 133).

[400] Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification (see Sec. 5.5.3 on page 136).

[401] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

[402] In German termed 'Perpetuierungsfunktion' or 'Integritätsfunktion' [147, 149].

[403] Namely, a mechanism offering ACA – UCD integrity would by definition not need to reliably detect authorized modifications. By definition, because this thesis did not aim to construct such a scheme as the property is not in the interest of an increased probative value.

[404] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[405] See Sec. 4.5.1 for a list of the signature functionalities from BT-Drs 14-4987 [147].

In the following this thesis bases its argumentation on the functional description of signatures that is considered the German policy makers' official reasoning on the German signature legislation implementing Directive 1999/93/EC [147]. These descriptions are helpful as they are presenting the goals of digital signatures in technology-neutral language: BT-Drs 14-4987 [147][405] and BT-Drs 17-10702 [149] are used in the following.

The functionality of identification demands that authentication MUST be verifiable in a non-interactive manner; only then it can be clearly argued that it is 'easy' or 'simple' for the Verifier to carry out the identification function[406]. The identification functionality states that the signature enables the Verifier to check with ease[407] whether the signature was generated using the signatory's secret signature generation key.

The demand for non-interactive public authentication of origin is also in line with another signature's functionality that requires that the hand-written signature can be checked by comparing it to another one of known authenticity using handwriting examination which does not require the active involvement of the signatory[408] as postulated by the German official reasoning in BT-Drs 17-10702 [149]. This second required functionality is often referred to as verification functionality[409] and is closely related to the identification functionality[410]. Thus, requiring non-interactive public accountability is also in line and aids to achieve the above stated functionality[408] without the need for an involvement of the signatory as postulated in BT-Drs 17-10702 [149].

With non-interactive public accountability (PUB) the identification function from [149] is fulfilled regardless of the cooperation of the Signer or the Sanitizer. This is also the case for the existing and already legally recognised digital signature schemes based on asymmetric cryptography — denoted by CDSS for classical digital signature scheme in this thesis. With asymmetric cryptography the integrity mechanism — with some additional infrastructure — can be used to identify the Signer and thus achieve a second required functionality: verification[409]. CDSS achieve the non-interactive public verification of origin. Namely, CDSS provide 3^{rd}-party accountability, which is in line with public verifiability and transferability (see also Definition 101). In this regard this thesis notes the following:

MSS and CDSS can be aligned till the only difference is ACA versus NCA. While there is always only one originator in CDSS — the Signer — a mechanism that offers the non-interactive and public form of authentication (PUB) also allows to publicly identify the origin. Thus such a feature technically aligns MSS and CDSS. This thesis seeks to construct MSS such that this alignment can be done. As a result there is finally only one remaining technical difference in the mechanisms' integrity protection: MSS offer ACA while CDSS offer NCA. See Sec. 19.1 for the overall conclusion.

7.3.3. Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity)

A mechanisms that offers ≥1CD – PUB integrity obviously achieves the legal verification function following [147, 149][411] notwithstanding the cooperation of the Signer, the Sanitizer, or any third party that has access to secret keys. This function is described to have the goal of binding content and signature to allow to conclude that the content was issued by the signatory[412]. Hence, the signature must achieve that the content shall not be subsequently modified[413]. The legal verification is closely related with

[406] In German it states "[...] kann der Adressat in **einfacher Weise** überprüfen, ob die Signatur **mit dem privaten Signaturschlüssel des Schlüssel-Inhabers erstellt** worden ist [...]" [147]; bold face has been added for highlighting.

[407] The phrase 'with ease' is used as the English correspondence to the German "in einfacher Weise" from [147].

[408] In German: "Die Identität kann im Streitfall z. B. durch einen Unterschriftenvergleich verifiziert werden." [149, p. 17].

[409] "Verifikationsfunktion" [147].

[410] "Die Verifikationsfunktion steht im engen Zusammenhang mit der Echtheits- und der Identitätsfunktion." [149, p. 16].

[411] See Sec. 4.5.1 for a list of the signature functionalities from BT-Drs 14-4987

[412] "Die räumliche Verbindung der Unterschrift mit der Urkunde, die den Erklärungstext enthält, stellt einen Zusammenhang zwischen Dokument und Unterschrift her. Hierdurch soll gewährleistet werden, dass die Erklärung inhaltlich vom Unterzeichner herrührt." [147, p. 16].

[413] "[...] wird gewährleistet, dass die Erklärung inhaltlich vom Signierenden herrührt und nicht nachträglich verändert worden ist." [147, p. 17].

the authenticity functionality. With $\geq$1CD – PUB the Verifier can carry out the validation of integrity with more ease than compared with INT, i.e., without any interaction with a party that has access to the Signer's secret key. Following the functionalities listed in Germany's codified interpretation of the legal texts this ease is important[414].

In this regard this thesis notes that also the existing and legally recognised asymmetric cryptography based digital signature schemes achieve the non-interactive public verification of integrity: Following the classification of RSASSA-PSS from PKCS-v2.2 [422] given in Sec. 6.10 (the visualisation of RSASSA-PSS from PKCS-v2.2 [422] is given in Sec. 6.10) $\geq$1CD – PUB is achieved by RSASSA-PSS from PKCS-v2.2 [422]. With a sufficiently large [74] security parameter RSASSA-PSS from PKCS-v2.2 [422] is legally accepted in Germany as it is listed in the German catalogue of algorithms (see Sec. 5.3.1).

7.4. Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer)

Following especially Analysis Result 4[415], any subsequent modification must be detectable by the integrity protection to be legally compliant. Thus, $\geq$1CD is required as the level of detection (as introduced in Sec. 6.3.1). Moreover, according to the Analysis Result 5[416] from the detection of a subsequent modification it follows that the mechanism MUST achieve accountability. This requires the generation of technical evidence such that it can be used to identify the origin. Technically this property is denoted as non-repudiation of origin. Hence, this thesis requires that the mechanism MUST achieve technical non-repudiation of origin (see Sec. 2.10.2.1). To be legally compliant a mechanism MUST make the Signer accountable towards the Verifier (see Analysis Result 5). On the same grounds as for Requirement 3 this thesis strongly recommends that the accountability SHOULD be verifiable in a non-interactive public manner. From this follows Requirement 4.1 (see Sec. 7.4.1).

Note, that accountability also for the Sanitizer is legally advised, but considered only optional (marked by using the keyword MAY) for an increased probative value after analysing Regulation 910/2014 as well as the former Directive 1999/93/EC. Achieving also non-interactive public accountability of the Sanitizer would provide the Verifier with additional legal guarantees towards the Sanitizer. The absence of the cryptographic property of transparency allows the Verifier to identify if there has been an authorized third party — the Sanitizer — that has generated the signature. This is captured as the optional Requirement 4.2 (see Sec. 7.4.2).

7.4.1. Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer)

Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer)
An integrity protection mechanism MUST

[414] In German BT-Drs 14-4987 states "[...] kann der Adressat in **einfacher Weise** überprüfen, [...] ob der signierte Text **nachträglich verändert worden** ist." [147, p. 17]; bold face has been added for highlighting.

[415] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

[416] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

1. *enable the Verifier to identify the presence or absence (1 bit information) of at least one occurred subsequent modification (≥1CD[a]) on the message (message-level[b].), both authorized and unauthorized, and*
2. *achieve at least interactive non-public correctly-executed and non-repudiable accountability (INT[c]) of the Signer given that no subsequent unauthorized modification(s) occurred, and*
3. *achieve technical non-repudiation of signature creation[d] by the Signer again given that no subsequent unauthorized modification(s) occurred.*

An integrity protection mechanism SHOULD

1. *achieve non-interactive public non-repudiable accountability (PUB[e]) of the Signer given that no subsequent unauthorized modification(s) occurred.*

[a] As introduced in Sec. 6.3.1.

[b] See Sec. 6.5 on page 156 and Sec. 13.2 on page 319 for further information on blocks- and message-level

[c] See Definition 110 on page 151.

[d] See Sec. 2.10.2.1 on page 34 and correctly-executed and non-repudiable INT as defined in Definition 110 on page 151.

[e] See Definition 112 on page 153.

The 'remote signature permission' of Regulation 910/2014 has been criticised for the missing option to distinguish between a Signer generated signature and a remotely generated one by *Roßnagel* [410]. Here, the capability of a MSS to generate a subsequent modification could be seen as a remote signature, i.e., a signature done with authorization of the Signer but not by the Signer. However, it is not the same as the input to a remote signature scheme is an unsigned message. In the same line, but based on the argument that any subsequent modification must be detected (following Analysis Result 4), this thesis postulates the following:

> The Verifier SHOULD be able to identify the presence or absence of at least one occurred subsequent modification from just the Signer's public verification key, the message and the signature.

Only then it can be clearly argued that it is 'easy' or 'simple' for the Verifier to carry out the identity function[417]. On the contrary, the opportunity to achieve this signature functionality[418] for schemes that are transparent, i.e., interactively non-publicly accountable (INT[419]), is diminished: A signature scheme where authorized subsequent modifications stay hidden from the Verifier unless the Signer interactively agrees to make them detectable might not easily be found technically equivalent to legally accepted schemes and thus might not offer the same increased probative value. This is even problematic if additional mechanisms on top of the MSS would give correctly-executed and non-repudiable INT. This is also due to the fact that a run of the verification algorithm alone would not achieve the expected authenticity functionality, which allows the Verifier to assume that the content of the signed document originated from the signatory[420]. Finally, and by far the heaviest and most obvious argument, public forms of accountability can be found in the existing and widely legally recognised asymmetric cryptography based digital signatures (CDSS), like the newer RSASSA-PSS algorithms in PKCS#1 v2.2 [422]. RSASSA-PSS from PKCS-v2.2 [422] as a CDSS provides 3rd-party non-interactive public accountability (see Sec. 6.10), which is in line with the notions of public verifiability and transferability[421]. The CDSS also provides non-repudiation of the signature generation by the Signer, such that it can be used to achieve a non-repudiation service as defined in Definition 88 on page 121. Hence, it remains strongly suggested (marked by SHOULD) for the legal signatory seeking an increased probative value, to choose a suitable scheme or control the accountability for subsequent modifications in order to maintain technical equivalence with existing cryptographic algorithms for qualified/advanced electronic signatures. With non-interactive public accountability the identification function can be provided by the mechanism re-

[417] In German BT-Drs 14-4987 states "[...] kann der Adressat in **einfacher Weise** überprüfen, ob die Signatur **mit dem privaten Signaturschlüssel des Schlüssel-Inhabers erstellt** worden ist und ob der signierte Text **nachträglich verändert worden** ist." [147]; bold face has been added for highlighting.

[418] See Sec. 4.5.1 for a list of the signature functionalities.

[419] Interactive non-public accountability (INT) as provided by the core RSS and SSS functionality is defined in Definition 109, the explicitly correctly-executed and non-repudiable version in Definition 110 on page 151.

[420] In German it states "Hierdurch soll gewährleistet werden, dass die Erklärung **inhaltlich vom Unterzeichner herrührt**." [147]; bold face has been added for highlighting.

[421] Transferability is defined in Definition 101 on page 128 following *Katz and Lindell*.

gardless of the cooperation of the Signer or the Sanitizer. This is also in line with the stated functionality that a hand-written signature can be checked by comparing it to another one of known authenticity using handwriting examination[408] without the need for an involvement of the signatory as postulated in the description of signature functionalities from German legal text [149].

Note, still this thesis does not make this need for public non-interactive accountability a mandatory requirement (indicated by MUST), but only a strong recommendation (indicated by SHOULD). The recommendation is based on the analysis of classical digital signature schemes that are currently judged as being legally compliant. This thesis did not find any explicit legal or technical definition which does require non-interactive public accountability.[422]

7.4.2. Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer)

Accountability of the Sanitizer is not considered a hard requirement for an increased legal probative value as offered by existing CDSS. However, allowing another entity to modify certain well-definable parts of the integrity-protected document subsequently might complicate taking liability for the Signer. Analysis Result 7[423], as well as Analysis Result 12[424], finds that the Signer, if doing it with consent, will take the liability for the authorized subsequently modified document including all authorized subsequent modifications that any authorized Sanitizer might have done. This makes the Signer liable towards the Verifier in the external relationship[425]. This does not mean that the Sanitizer might not be liable towards the Signer in the internal relationship[426]. To make this liability of the Sanitizer for any authorized subsequent modifications cryptographically visible to the Verifier, the Signer will need to use an integrity protection mechanism which allows subsequent authorized modifications only if the mechanism additionally also (1) identify the Sanitizer once subsequent modifications have occurred as the source of the modified document and (2) make the Sanitizer cryptographically accountable for occurred subsequent modifications. Also, *Roßnagel* in [410] — citing [147, 149] — states that the identification function is no longer provided if the person generating the signature is not the entity that controls the signature generation key, i.e., the Signer. [427] Having the Sanitizer non-interactively publicly accountable for an occurred subsequent modification allows the Verifier to appoint the Sanitizer as the origin, which weakens the argument that the identity function cannot be fulfilled. Furthermore, if the Sanitizer is appointed the originator the expected authenticity functionality, which allows the Verifier to assume that the content of the signed document originated from the defined and accountable party of the Sanitizer, which was endorsed by the original signatory[428]. With having the Sanitizer be non-interactively publicly accountable for the documents — or block(s) thereof with block-level accountability — it can be argued to be treated as a Signer in its own right. See also the argumentation provided in the final legal evaluation especially in Sec. 17.5.2.4. This is captured in the following:

Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer)

An integrity protection mechanism with ACA[a]) MAY, given that at least one subsequent authorized modification occurred and no subsequent unauthorized modifications occurred,

1. *achieve non-interactive public accountability (PUB[b]) of the Sanitizer,* ***and***
2. *achieve technical non-repudiation of the occurred subsequent modification by the Sanitizer.*

[a] as defined in Sec. 6.2.3

[b] as defined in Definition 112

[422] Reconsider Sec. 6.8 with the analysis overview of the accountability of existing technical definitions.

[423] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[424] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

[425] In German 'Aussenverhältnis'.

[426] In German 'Innenverhältnis'.

[427] In German: "Sie [die Identitätsfunktion] ist jedoch nicht mehr gewährleistet, wenn "die Person des tatsächlich Signierenden nicht mit der des Signaturschlüsselinhabers überein(stimmt)" [147, 149]" [410]

[428] In German it states "Hierdurch soll gewährleister werden, dass die Erklärung **inhaltlich vom Unterzeichner herrührt**." [147]; bold face has been added for highlighting.

7.4.3. Requirement 4 in the Multi-Sanitizer settings

This thesis understanding of multiple sanitizers is based on the formal definition by *Canard, Jambert, and Lescuyer* [112]:

> "Our aim is to propose a model where one signer (among n) can choose a set of sanitizers (among m) such that any sanitizer of the chosen subset is able to sanitize the output message-signature pair. Moreover we want to be consistent with the initial model from Brzuska et al. [64] where one signer chooses one sanitizer." [112].

This thesis will call a setting in which the MSS allows to distinguish between two or more Sanitizers a *multi*-Sanitizer *setting*. In this setting the MSS schemes could still offer accountability notions that appoint one ore more Sanitizer(s). Thus, requirements for accountability and non-repudiation of Sanitizer(s) can be met in the multi-Sanitizer setting as well.

Multi-Sanitizer-accountability MUST appoint involved Sanitizer(s) via their public key(s) whenever a protected message or blocks of it are modified in an authorized way. For the message-level accountability a compliant mechanism in the multi-Sanitizer setting MUST allow the Verifier to — either interactively (INT) or non-interactive publicly (PUB) — identify all involved Sanitizer(s) for each Sanitizer that actually influenced the subsequently modified message in question under the assumption that at least one modification occurred and that only authorized subsequent modifications were done.
For the block-level accountability a mechanism in the multi-Sanitizer setting MUST allow the Verifier either interactively (INT) or non-interactive publicly (PUB) to identify the one Sanitizers that was involved in the last subsequent modification to the block in question under the assumption that at least one modification occurred to that block and that only authorized subsequent modifications were done to that block.
Note, in both cases if the same block is modified repeatedly by different Sanitizer(s) only the last modification will be taken into account by the accountability notion in RSS and SSS schemes. This is due to the offered privacy protection which overrides and hides any previous modifications.
Further, the accountability and non-repudiation in the multi-Sanitizer framework MUST allow to create evidence that point at the legal entity related to the Sanitizer's cryptographic key.
If all the above are met, the multi-sanitizer framework does not introduce new legal obstacles.

This thesis does not consider the multi-Sanitizer setting, its listed as future work in Sec. 19.3.

7.4.4. Link to the legal signatory is out of scope of Requirement 4

Note that the above technical accountability requirement establishes technical evidence for an accountability of an entity being in the role Signer (or respectively Sanitizer). Entities can take this role due to their knowledge of a corresponding secret, as defined in Sec. 3.6.1. However, legal accountability towards the signatory, i.e. the natural person[429], has been demanded as Analysis Result 5[430] and Analysis Result 7[431] have yielded. Hence, for full legal recognition the technical accountability mechanism MUST be able to point at the signatory, i.e., the natural person in the role of the Signer[432]. The Requirement 5, introduced next, will capture this separately: This thesis has chosen to keep them separate, as the accountability requirement expressed in Requirement 4.1 MUST be fulfilled with the cryptographic signature scheme, while the bond between the legal or natural person is not in the scope of the signature scheme[433], but relates more to the management of public key certificates (see Definition 26 in Sec. 2.11 on page 35).

[429] As defined in Definition 27.

[430] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[431] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[432] Note regarding term legal person in Regulation 910/2014: "signatures" are created by a natural person and "seals" are created by a legal person.

[433] Linking the legal signatory and the technical signature value is also out of scope of the thesis, see Sec. 1.4.3.

7.5. Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate)

This thesis separates the technical mechanisms for accountability by giving also two requirements, as discussed in Sec. 7.4.4. The previous Requirement 4 prescribes a set of mechanisms that links a message to a Signer's (or respectively Sanitizer's) — denoting a technical role — public signature verification key to establish accountability. This Requirement 5 is for another set of mechanisms that provide the linkage between a public key and the signatory — denoting a natural (or legal) person — as legally required following from Analysis Result 5[434] and Analysis Result 7[435]. From the viewpoint of only the cryptographically secure digital signature schemes, the fulfilment of from Analysis Result 5[434] and Analysis Result 7[435] is requiring an additional set of mechanisms and organisational rules. Requirement 5 captures that it is legally required that a digital signature scheme supports the usage of such additional mechanisms; because of their capability to support this, digital signature schemes based on asymmetric cryptography are at the technical focus (following Analysis Result 13[436]). Finally, related to Analysis Result 12[437] this requirement provides the link to the legal signatory, as technically digital signature mechanisms through signature verification identify only that the public signature verification key corresponds to the secret key involved in the signature's generation.

The integrity protection's accountability and non-repudiation towards the signatory described in the legal texts builds upon the notion of qualified public key certificates (see defined in Definition 26). In a nutshell, the legal integrity protection requires that a signature must be "linked to the data to which it relates in such a manner that any subsequent change of the data is detectable" [175]. Further, for authentication of origin (authenticity), the corresponding public keys must be certified by certificate authorities, i.e., "uniquely linked" [175] and "capable of identifying the signatory" [175].

Following that, this thesis defines the following technical requirement:

Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate)
A legally accountable integrity protection mechanism MUST *emit a public signature verification key, that allows verifying the involvement of a corresponding secret signature generation key during signing. This public verification key* MUST *enable a trusted third party to issue a qualified certificate to link the verification key to a natural (or legal person[a]).*

[a] This thesis kept the reference to the legal person here due to the definitions of an electronic "seal" defined for legal persons and an electronic "signature" defined for natural persons in Regulation 910/2014 are from their technical description very close; note that the consequences for probative value are however different as discussed in Sec. 4.4.3.

Fulfilment of Requirement 5 then aids the identification functions (see Sec. 4.5.1). In [147, 149] it is described as being fulfilled that a unique asymmetric key-pair can be appointed to a natural person[438].

7.5.1. Key management is part of Requirement 5, but details are out of scope of this thesis

This thesis does not go into further details on how the public key's qualified certificates are technically built, how a public key infrastructure needs to be constructed, nor discuss if this is the right technical solution. See the thesis of *Brands* [63] or discussions by *Gutmann* [220] for further ideas. In order to do this, this thesis made the following two security assumptions about the cryptographic keys in order to be able to base authentication of legal persons on asymmetric cryptography:

[434] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[435] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[436] Analysis Result 13: Digital signature schemes are a suitable integrity and authenticity protection mechanism (see Sec. 5.5.6 on page 140).

[437] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

[438] In German: "Identitätsfunktion [...] dadurch erfüllt, dass ein jeweils einmaliges Signaturschlüsselpaar durch anerkannte Stellen einer bestimmten natürlichen Person zugeordnet wird" [147, 149].

- Security Assumption 1 (Secret keys are not known to the adversary) (see Sec. 3.8.2)
- Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) (see Sec. 3.8.3)

Especially, the public key distribution must not only be working, but for legal reasons also make use of legally qualified electronic public key certificates, as defined in Definition 26 in accordance with the Regulation 910/2014. Thus, Requirement 5 does not prescribe that the integrity protection mechanisms must technically and procedurally achieve the legal requirements by itself, but rather that it must technically support the creation of a public key certificate that fulfils all the legal requirements in order to bind the electronic signature to a natural person.

In order to build upon such existing technical solutions the mechanism needs the ability to expose a public signature verification key.

7.5.2. Notes on achieving Requirement 5

Technically, Requirement 5 demands to build the mechanism upon asymmetric cryptography in order to expose a public signature verification key for inclusion in a public key certificate. *Laborde* noted this already when stating that the directive [175] "implicitly" [303] endorses cryptographic digital signatures by "[...] establishing requirements that, so far, can only be fulfilled by using digital signatures [...]" [303]. Existing technical solutions, so-called public key infrastructures (PKI), are built upon public key certificates adhering to the X.509 standard [236, 248] and Certificate Issuers that are trusted identity providers following Regulation 910/2014, e.g. the providers listed in [170]. Then the above legally accepted PKI can fulfil the legal requirement of binding the electronic signature to a natural person as required in Requirement 5. In detail, the information contained in the qualified public key certificate must be the MSS public signature verification key of as well as the identifier for the legal entity as with CDSS. Finally, a correctly working certificate issuer means that it vouches that it followed all the rules resulting from the legislation, i.e., in the EU the Directive 1999/93/EC or the Regulation 910/2014.

7.6. Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD))

Following the legal Analysis Result 6[439] stated in Sec. 4.6.6 the signature generating algorithm that uses the secret signature generation key $\mathsf{sk}_{\mathrm{sig}}$ must be executed in an environment that has some additional properties. In Regulation 910/2014 this execution environment made from hardware and software is called a "qualified electronic signature creation device" [Regulation 910/2014] or QSCD in short. The previous Directive 1999/93/EC referred to this as a "secure-signature-creation device" [Directive 1999/93/EC], or SSCD. The QSCD requirements are in Annex II Regulation 910/2014. The analysis from Sec. 4.4.7 has shown that there are only marginal differences between a QSCD and an SSCD. Especially, the differences do not influence the probative value in the light of authorized subsequent modifications.

Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD))
The integrity protection mechanism MUST *enable the legitimate signatory, i.e., technically the entity in the role of the* Signer*, by appropriate technical and procedural means:*

1. *to keep the secret signature generation key (sk_{sig}) confidential, and*
2. *to be able to ensure that sk_{sig} an practically occur only once, and*
3. *to protect it against the use of others.*

[439] Analysis Result 6: Qualified electronic signatures on electronic documents when computed by qualified electronic signature creation device (QSCD) award the same probative value that handwritten signatures award to paper documents following member state rules (see Sec. 4.6.6 on page 106).

7.6.1. Notes on achieving Requirement 6

Requirement 6 can be achieved by involving a hardware security module (HSM) as defined in Definition 29. Being an HSM allows the QSCD/SSCD to use — and to also generate — the secret keys while keeping them confidential. This is how currently the legally accepted public key infrastructures operate: There is a Common Criteria [467] protection profiles specified for a “Trustworthy Signature Creation Module (TSCM) used by TSP [Trusted Service Provider] as part of their infrastructure [...] that generates advanced electronic signatures as defined in Regulation (EU) N°910/2014” [468]. Apart from the HSM usable by trusted service providers other common criteria protection profiles exists for QSCD or SSCD. There a SSCD/QSCD is defined as “[...] a combination of hardware and software configured to securely create, use and manage signature-creation data [...]” [466]. Further, especially QSCD get accredited under Regulation 910/2014 Article 31 (1)-(2) through EU member states (seey [171] for a compiled list of those QSCD).

7.6.2. Relations between Requirement 6 and other requirements and the thesis security assumptions

Requirement 6 technically supports the Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) (see Sec. 3.8.3) as follows: If the mechanism is able to support Requirement 6 for the Signer then the confidentiality assumption that the Signer is the only entity who can use the secret $\mathsf{sk}_{\mathrm{sig}}$ to generate valid signatures for documents holds. Respectively, if the mechanism requires a QSCD/SSCD for the Sanitizer then the Sanitizer is the only entity who can use $\mathsf{sk}_{\mathrm{san}}$ to generate valid signatures for correctly sanitized documents. Hence, Requirement 6 operationally supports achieving the accountability that was required in Requirement 4[440] and follows from Analysis Result 5[441] and Analysis Result 7[442].

Additionally, Requirement 6 enables to support the requirements 1 (a), (b) and (d) of Annex II Regulation 910/2014. Annex II Regulation 910/2014 was analysed in Sec. 4.4.7. The following Requirement 7 will cater for the left out requirement 1 (c) of Annex II Regulation 910/2014.

7.7. Requirement 7 (Mechanism MUST achieve a legally accepted state-of-practice strength of the cryptographic algorithm used for integrity and authenticity protection)

To be usable, the mechanism must suitably prevent unauthorized actions from happening to the integrity-protected documents without it being reliably detected. This follows from Analysis Result 5[441], Analysis Result 4[443]; and it is required to comply with the technical Analysis Result 10[444]. Any unauthorized and undetected subsequent modification on an integrity-protected message must therefore be prevented. Those unauthorized and undetected subsequent modifications are denoted by the term *forgery*. The strength against such a forgery is described in relation to the current adversaries’ capabilities. As identified in Analysis Result 6[445], requirement 1 (c) of Annex II Regulation 910/2014 states that “[...] the electronic signature is reliably protected against forgery using currently available technology [...]” [Regulation 910/2014][446].

[440] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[441] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[442] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[443] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

[444] Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification (see Sec. 5.5.3 on page 136).

[445] Analysis Result 6: Qualified electronic signatures on electronic documents when computed by qualified electronic signature creation device (QSCD) award the same probative value that handwritten signatures award to paper documents following member state rules (see Sec. 4.6.6 on page 106).

[446] Note that Regulation 910/2014 added “reliably” in comparison to Annex III of Directive 1999/93/EC.

Requirement 7 (Mechanism MUST achieve a legally accepted state-of-practice strength of the cryptographic algorithm used for integrity and authenticity protection)
The integrity protection mechanism MUST *reliably protect against a forgery (any occurred unauthorized modification to integrity-protected data that is not reliably detected). Reliably means that the mechanism* MUST *be*

1. *considered cryptographically secure against state of the art attacks, and*
2. *that the algorithm must be legally accepted.*

Requirement 7 contains a cryptographic and a legal component. The thesis highlights this by splitting the requirement into three sub-requirements:
Firstly, Requirement 7.1 presents the notion of a forgery and the definition of cryptographic strength based on the existing analysis of suitable cryptographic definitions.
Secondly, Requirement 7.2 lists the sources for technical information from which legal texts deduct that they are legally suitably strong enough to induce a probative value at the level of prima facie evidence or that of evidence with a legal presumption of authenticity (see the statutory rules in the ZPO discussed in Sec. 4.2.2). For a new algorithm this thesis sees it as a pre-requisite for a legal recognition that the new algorithm achieves at most the same cryptographic strength as current ones. For example, while a cryptographic viewpoint taken in Requirement 7.1 might favour algorithms to be quantum-safe, it might not yet be a legal requirement.
Finally, Requirement 7.3 (opt.) is the optional requirement that the algorithm MAY be listed in the technical documents used as information sources identified by legal texts to deduct that this specific algorithm is legally suitable. This would then provide an increased probative value with legal certainty[447]. The requirement is kept optional on purpose because the lists are intended to change over time when technical developments and standards are updated. The signature laws of Europe have been drafted to be technology-neutral[448]. The requirements marked as mandatory in this thesis are those that correspond to a demand stated in legal texts and thus a failure to achieve them would disobey the law[449].

7.7.1. Requirement 7.1 (Mechanism MUST achieve cryptographic protection against a forgery)

The following is a comparison between CDSS and MSS definitions of the cryptographic attack of a forgery. With the term forgery of a digital signature this thesis by definition means the existential forgery under an adaptive chosen message from the attack defined by *Goldwasser et al.* in 1988 [211] (see Sec. 5.3.4). A digital signature scheme that is existentially unforgeable under adaptive chosen message attack (EUF-CMA) allows detecting forgeries of the signed message done by an adversary with no access to the secret key (see Definition 17 for a description). Looking more closely, the formal Definition 95 of existential unforgeability under an adaptive chosen message attack (EUF-CMA) restated below classifies all validly signed messages which have not been created using the Signer's sk_{sig} as valid forgeries. Technically, to adapt to a change in the strength of the current adversaries' capabilities would be achieved by adjusting the security parameter, denoted λ in the formal definitions below, for generally cryptographically secure cryptographic algorithms.

For MSS, however, the EUF-CMA security is too strong because MSS introduce the special case of authorized modifications, which shall be no forgeries, and which are generated validly by Sanitizers without access to the Signer's key sk_{sig}. For ease of readability this thesis restates the formal Definition 95 given in Sec. 5.3.4:

Definition 95 : Existential Unforgeability under Chosen Message Attack (EUF-CMA) following *Goldwasser et al.*

A signature scheme S *consisting of three algorithms with polynomial runtime (PPT)* $\mathsf{S} = (\mathsf{KGen}_{sig}, \mathsf{Sign}, \mathsf{Verify})$ *is said to be existentially unforgeability under chosen message attack (EUF-CMA), if for any efficient (PPT) adversary* $\mathcal{A}$ *the probability that the experiment* $\mathsf{Unforgeability}_{\mathcal{A}}^{EF-CMA}(\lambda)$ *given in Fig. 28 returns* 1*, is negligible (as a function of* λ*).*

[447] As defined in Definition 111 on page 153; see Sec. 6.4.2 for a discussion of the term.
[448] See Analysis Result 3.3 in Sec. 4.6.3.3 on page 104.
[449] Or where the law still has room for interpretation it would oppose the legal scholars' current standing interpretation or the courts' current application of the law.

```
**Experiment** $\mathsf{Unforgeability}_{\mathcal{A}}^{EF-CMA}(\lambda)$
  $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^\lambda)$
  $(m^*,\sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{pk})$
    for $i = 1,2,\dots q$ let $m_i$ and $\sigma_i$
      denote the adaptive queries and answers to and from the oracle Sign
  return 1 iff
    $\mathsf{Verify}(\mathsf{pk},m^*,\sigma^*) = 1$ and
    $\forall i, 1 \leq i \leq q : m^* \neq m_i$
```

Figure 28. Existential unforgeability under chosen message attack (EUF-CMA) Experiment (same Figure as on page 126)

```
**Experiment** $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\mathsf{sig}},\mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\mathsf{san}},\mathsf{sk}_{\mathsf{san}}) \leftarrow \mathsf{KGen}_{\mathsf{san}}(1^\lambda)$
  $(m^*,\sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk}_{\mathsf{sig}},\cdots)\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathsf{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathsf{san}})}(1^\lambda,\mathsf{pk}_{\mathsf{sig}},\mathsf{pk}_{\mathsf{san}})$
    for $i = 1,2,\dots q$ let $(m_i,\mathsf{ADM}_i,\mathsf{pk}_{\mathsf{san,i}})$ and $\sigma_i$
      denote the adaptive queries and answers to and from the oracle Sign
    for $j = q+1,\dots,r$ let $(m_j,\mathsf{MOD}_j,\sigma_j,\mathsf{pk}_{\mathsf{sig,j}})$ and $(m'_j,\sigma'_j)$
      denote the adaptive queries and answers to and from the oracle Sanit
  return 1, if
    $\mathsf{Verify}(1^\lambda,m^*,\sigma^*,\mathsf{pk}_{\mathsf{sig}},\mathsf{pk}_{\mathsf{san}}) = \mathtt{true}$ and
    $\forall i = 1,2,\dots,q : (\mathsf{pk}_{\mathsf{san}},m^*,\mathsf{ADM}^*) = (\mathsf{pk}_{\mathsf{san,i}},m_i,\mathsf{ADM}_i)$ and
    $\forall j = q+1,\dots,r : (\mathsf{pk}_{\mathsf{sig}},m^*,\mathsf{ADM}^*) = (\mathsf{pk}_{\mathsf{sig,j}},m^*,\mathsf{ADM}_j)$
```

Figure 47. Unforgeability Experiment for SSS based on *Brzuska et al.* [64] and *Gong et al.* [216] (same Figure as on page 261)

To see the difference, assume an MSS is used instead of a CDSS for the signature scheme S in Definition 95. With an MSS the Sanitizer can subsequently modify a message m into m^+ and adapt the signature it got from the Signer into σ^+. The pair (m^+,σ^+) will verify under the Signer's key, assuming that the Signer generated (m,σ) with an MSS and that the Sanitizer only carried out authorized modifications to produce m^+ from m. Such a valid sanitization or redaction can thus yield a *fresh* message according to Definition 95. Fresh means m^+ was never signed by the Signer or the signing oracle available to the adversary, or formally $\forall i, 1 \leq i \leq q : m^+ \neq m_i$ where $i = 1,2,\dots,q$ index the adaptive queries to the signing oracle. Hence, any valid sanitization or redaction creates a validly signed message that fulfils both requirements from the EUF-CMA Definition 95 game depicted in Fig. 28.

To overcome that any Sanitizer would break unforgeability MSS first adjusts the unforgeability property and second introduces the cryptographic property of immutability as an MSS equivalent of unforgeability.

The unforgeability property for MSS is re-defined from that for DSS such that it excludes from the category of forgeries those messages with their valid signatures that are created due to an authorized subsequent modification by Sanitizers. To recap, in DSS any subsequent modification is unauthorized and thus flagged as a forgery. In a MSS only an unauthorized subsequent modification with a valid signature is considered a forgery. This is a security goal of the Signer and its appointed Sanitizers towards Verifiers. In Sec. 11.6.1 this thesis thus accordingly gives the following high-level definition for SSS[450]:

Definition 151 : Unforgeability for SSS (high-level)

> ***Unforgeability*** *guarantees that a signature valid under* pk_{sig} *on a message can only be generated*
>
> - *by the* Signer *with access to* sk_{sig} *corresponding to* pk_{sig}*, or*

[450] For RSS unforgeability is given in Sec. 11.13.1.

- *by a Signer-endorsed Sanitizer with access to sk_{san} that is sanitizing a message-signature pair where the signature is valid under pk_{sig} and where all sanitizations are conforming to the Signer-endorsed modification policy (ADM).*

Note 1 All valid sanitizations are excluded from being forgeries.

Note 2 Unforgeability is as strong as possible; it shall explicitly prohibit unauthorized re-ordering of blocks of structured data, as well as mix-and-match attacks.

Note 2 in the definition of unforgeability is inspired by the remark of *Steinfeld et al.* [455][451] and by the aim to make the MSS as strong as possible in comparison to the CDSS.

The formal definition of unforgeability for SSS from Sec. 11.6.1 is as follows:

Definition 152 : Unforgeability for SSS

*An SSS is **unforgeable**, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment* $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ *depicted in Fig. 47 returns* 1*, is negligible (as a function of* λ*).*

The adversary is in the role of the Verifier and does not possess any access to secret keys. However, it can adaptively generate validly signed messages of its own choice by accessing now two oracles; the additional one is the Sanit oracle. As before, the adversary will not break unforgeability when the verifying attack message m^* has been given to the oracle for signing. Differently to EUF-CMA Definition 95 in the SSS unforgeability notion — based on *Brzuska et al.* [64] and *Gong et al.* [216] — the adversary only wins the game if the message was also not previously given to the Sanit oracle. By this means, Definition 152 excludes sanitizations from the set of forgeries.

The immutability property has been introduced by *Ateniese et al.* in [12]. In a nutshell, immutability describes which subsequent modifications are never authorized. The Signer defines the block(s) of the integrity-protected message that are **not** allowed to be subject of any subsequent modification, not even an authorized modification. Those block(s) are thus termed *immutable*. Immutability is a security goal of the Signer towards Sanitizers. It is equal to the standard signature's unforgeability notion which describes the Signer's target to be achieved towards Verifiers. In Sec. 11.6.2 this thesis gives a formal definition of immutability for SSS[452] as well as the following high-level definition:

Definition 153 : Immutability for SSS (high-level)

***Immutability** prevents a Sanitizer from deriving a verifying signature after modifying any block(s) not marked as admissible for this Sanitizer by the Signer.*

7.7.2. Requirement 7.2 (Mechanism MUST be as good as the legally accepted current state of practice for integrity and authenticity protection)

Sec. 5.3 identified recommendations and standards that list what current cryptographic algorithms and key lengths are currently legally accepted. The algorithms and the construction used in an MSS MUST achieve at least the same strength as those algorithms for electronic signatures that are currently legally recognised to become recognised themselves. Note, the moment sophisticated attacks are known to a scheme, or an attack reduces the effective protection of a specified key length, all recommendations (ENISA [188], Germany's catalogue of algorithms [85], and NIST SP 800-131A [25]) do no longer encourage the use of this key length or of this algorithm. Hence, no attacks on the MSS' primitives and the used security parameter for the generated keys must be known to achieve Requirement 7. Finally, the argumentation for an MSS offering an equivalent strength becomes easier if the underlying cryptographic primitives embodied in MSS are already legally recognised and assumed to be still strong primitives.

[451] "We remark that the definition of CES-unforgeability above is as strong as one may hope for in our setting, i.e. it prevents forgeries for any documents which are not subdocuments of signed documents. This implies, in particular, security against 'submessage reordering attacks' (where the submessages in the forged document are the same as those in subdocument of a signed document but in different positions or order) as well as 'mixed subdocument attacks' (where the forged document contains submessages which have all appeared in signed documents but have never appeared together in a signed document)."[455]. This thesis uses the term 'mix-and-match attack' to refer to the latter attack.

[452] Immutability for RSS is defined in Sec. 11.13.2.

As an example take the SSS construction $pubacc\mathcal{SSS}$ from Sec. 13.4, which is built upon two interwoven DSS. $pubacc\mathcal{SSS}$ can be instantiated using RSASSA-PSS algorithms [422] (see Sec. 13.5). Hence, the building blocks of $pubacc\mathcal{SSS}$ are legally recognised mechanisms[453].

EXAMPLE

Regarding key management Requirement 7 includes the requirement that the asymmetric key pair is generated securely and distributed correctly. However, the details of this key management are out of scope. Instead, for the generation of keys this requirement assumes that all keys are generated using sufficiently large security parameters as prescribed by the most current legally accepted recommendations and standards. Further, this thesis assumes that the algorithms for key generation are executed correctly. Combined with the requirement that all parties keep their secret keys confidential and prohibit unauthorized use of them this leads to the assumption explicitly stated as Security Assumption 1 (Secret keys are not known to the adversary)[454]. Further this thesis stated the exclusion of key distribution as Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed)[455]. This exclusion is practical and commonly made, e.g., in ISO 13888-1 [243] which looks at evidence generation mechanisms but also excludes the handling of verification keys[456].

7.7.3. Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms)

As analysed in Sec. 5.3 what current cryptographic best practice is legally accepted is based on recommendations and standards issued by several organisations on a regular basis. The algorithms for electronic signatures fall in the following categories: hash-functions, asymmetric digital signature schemes, and key generation.[457] In the domain of digital signatures this thesis looked specifically at the the following three technical sources:

- Germany's *catalogue of algorithms* [84] (see Sec. 5.3.1),
- ENISA's *Algorithms, key size and parameters report* [188] for the EU (see Sec. 5.3.2), and the
- *NIST Special Publication 800-131A* [25] for the United States of America (see Sec. 5.3.3).

Those prescribe suitable cryptographic algorithms covering all three categories as well as large enough security parameters for each of them. If the MSS algorithms would already be registered there, then it would be legally assumed that the technical mechanisms provide the legally required strength against state-of-the art attacks (see Sec. 5.3). It is kept optional (indicated by MAY) due to the technology-neutral language used in the texts of the European electronic signature legislation describing the intended outcome rather than a specific technology (see Sec. 4.6.3.3), as an example see the legal acceptance of the so-called container signature (see Sec. 4.5.2).

7.8. Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional))

Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional))
The integrity protection mechanism MAY *allow the* Verifier *to link two messages m' and m'' with their valid signatures whenever they have been created by valid redactions or sanitizations from the same m.*

In this regards, this thesis notes that in many RSSs in the literature the possibility of *merging* two redacted documents is implicitly present. The notion and definition of an algorithm for the action merge is given in this thesis for the scheme $merge\mathcal{RSS}$ in Sec. 14.11:

[453] Can be instantiated using algorithms listed as suitable till 2022 in the 2016 edition of German catalogue of algorithms [85].

[454] See Sec. 3.8.2.

[455] See Sec. 3.8.2.

[456] Namely, ISO 13888-1 states that "[t]he provision of the public verification key to the verifier depends on the type of signature scheme applied to generate the digital signature." [243].

[457] See also [Anlage 1 Abschnitt I Nr. 2, SigVO], and the catalogue of algorithms [84].

> "given two set-signature pairs derived from the same signature, one can combine them into a single *merged*, set-signature pair $(\mathcal{S}, \sigma)$" [387]

A successful merge would be a test for linkability. By merging or linkability one can detect duplication of documents that are generated by redaction and still treat them as one single evidence and not as multiple ones, e.g., if filed for the legal body of evidence. This would ease to use a linkable but less sanitized version in order to gradually reveal information that was previously confidentiality protected but keeping the chain of custody intact. This makes it useful to continue a chain of evidence and aid the evidentiary function as foreseen to be fulfilled by electronic signatures as discussed in Sec. 4.5.1. The opposite property is termed unlinkability in SSS literature; Sec. 11.6.5 gives the cryptographic definition for SSS based on existing literature. Further to this, the thesis offers a description of the algorithms required for linkability in Sec. 14.2.2; the idea of explicit instead of implicit linkability and the corresponding algorithms for RSS were also published as a technical report [393] (see Appendix A publication n^{o} 25).

In the application example for RSS presented in Sec. 18.4 linkability increases the legal usefulness of the application of RSS in a supply chain. The result of this thesis that linkability between an original and a redacted document is legally useful was also published [382] (see Appendix A publication n^{o} 9).

7.9. Relation between technical requirements for increased probative value and analysis results

Following the methodology[458], the analysis results have influenced the technical requirements. Some Analysis Results are more general in their nature: The legal Analysis Results 1-3[459] as well as the technical Analysis Result 11[460] show that the legal texts and technical definitions do neither disaffirm nor generally reject the idea of authorized subsequent modifications, which are central to MSS. In the same manner, Analysis Result 13[461] is also considered general as it supports the chosen focus on digital signatures as a technical enforcement. Those general results are omitted in the following tables.

The Analysis Result 3.3 yielded that German legal texts also allow to argue for an increased probative value based on functional equivalence (see Sec. 4.5.1 for a list of those functions considered to be fulfilled by a signature in general). The requirements where designed to incorporate these functionalities, they are listed in Tab. 6 separately. Further Analysis Result 3.3 showed[462] that it is possible to gain legal recognition on a case-by-case basis even if the exact algorithm is not listed, this is mirrored by Requirement 7.3[463] being separated and optional. The result is a '✓' in Tab. 4.

For all other Analysis Results towards an increased probative value the following tables have been composed: If an analysis result has been taken up by a requirement to increase the probative value it is marked by a '✓' in Tab. 4 for the analysis of legal sources and in Tab. 5 for those stemming from technical sources.

[458] Related to step 1 (Distill requirements (including legal) for the application domain(s)) and step 2.2 (Formalise application requirements (including legal) using technically precise terminology) of the methodology described in Sec. 1.5.

[459] Analysis Result 1: Legal texts acknowledge the importance of integrity.
Analysis Result 2: Legal acts do not explicitly forbid authorized modifications.
Analysis Result 3: European and German legal texts discuss the rules for probative value and statutory evidentiary presumption for digital/electronic documents; and they point to electronic signature legislation which describe the requirements for qualified/advanced electronic signatures in technology-neutral language.

[460] Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect (see Sec. 5.5.4).

[461] Analysis Result 13: Digital signature schemes are a suitable integrity and authenticity protection mechanism (see Sec. 5.5.6 on page 140).

[462] Using also the German court case on 'container signatures' (BGH Beschluss vom 14. Mai 2013 – VI ZB 7/13 [81]) discussed in Sec. 4.5.2 on page 96.

[463] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms).

	Legal Analysis Results (except 1-3.2)				
Tech. requirements for an increased probative value	Analysis Result 3.3: Electronic signature legislation texts describe required functionality in technology-neutral language	Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected	Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory	Analysis Result 6: Qualified electronic signatures on electronic documents when computed by qualified electronic signature creation device (QSCD) award the same probative value that handwritten signatures award to paper documents following member state rules	Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights
Requirement 1 [464]	–	√	–	–	√
Requirement 2 [465]	–	–	√	–	√
Requirement 3.1 [466]	–	√	–	–	–
Requirement 3.2 [467]	–	√	√	–	–
Requirement 3.3 (rec.) [468]	–	–	–	–	–
Requirement 4.1 [469]	–	√	√	–	–
Requirement 4.2 (opt.) [470]	–	–	–	–	(√) [471]
Requirement 5 [472]	–	–	√	√	√
Requirement 6 [473]	–	–	√	√	√
Requirement 7.1 [474]	–	√	√	√	–
Requirement 7.2 [475]	–	√	√	√	–
Requirement 7.3 (opt.) [476]	√	–	–	–	–
Requirement 8 (opt.) [477]	–	–	–	–	–

Table 4. Relation between legal analysis results and technical requirements elicited in this thesis for an increased probative value; *rec.=recommended*; *opt.=optional*

[464] Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer)

[465] Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier).

[466] Requirement 3.1 (Mechanism MUST verify the absence of any subsequent modifications, i.e., ≥1CD integrity).

[467] Requirement 3.2 (Mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory).

[468] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity).

[469] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer).

[470] Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer).

[471] A mechanisms that achieves the public form of accountability (PUB) achieves this more obviously.

[472] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate).

[473] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)).

[474] Requirement 7.1 (Mechanism MUST achieve cryptographic protection against a forgery).

[475] Requirement 7.2 (Mechanism MUST be as good as the legally accepted current state of practice for integrity and authenticity protection).

[476] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms).

[477] Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional)).

		Technical Analysis Results (except 11 and 13)			
		Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law	Analysis Result 9: Definition of authorized and unauthorized modification; and modification in general	Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification	Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document
Tech. requirements for an increased probative value	Requirement 1 [478]	✓	✓	–	✓
	Requirement 2 [479]	✓	✓	–	✓
	Requirement 3.1 [480]	✓	✓	✓	✓
	Requirement 3.2 [481]	–	–	–	✓
	Requirement 3.3 (rec.) [482]	–	–	–	(✓) [483]
	Requirement 4.1 [484]	✓	✓	✓	✓
	Requirement 4.2 (opt.) [485]	–	✓	✓	(✓) [486]
	Requirement 5 [487]	–	–	–	✓
	Requirement 6 [488]	–	–	–	✓
	Requirement 7.1 [489]	–	–	✓	✓
	Requirement 7.2 [490]	–	–	✓	✓
	Requirement 7.3 (opt.) [491]	–	–	–	–
	Requirement 8 (opt.) [492]	–	–	–	–

Table 5. Relation between technical analysis results and technical requirements elicited in this thesis for an increased probative value; *rec.=recommended*; *opt.=optional*

[478] Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer).

[479] Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier).

[480] Requirement 3.1 (Mechanism MUST verify the absence of any subsequent modifications, i.e., ≥1CD integrity).

[481] Requirement 3.2 (Mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory).

[482] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity).

[483] A scheme which offers the public form of integrity detection (PUB) more obviously achieves this.

[484] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer).

[485] Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer).

[486] The public form of accountability (PUB) eases the argumentation for this Requirement being fulfilled.

[487] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate).

[488] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)).

[489] Requirement 7.1 (Mechanism MUST achieve cryptographic protection against a forgery).

[490] Requirement 7.2 (Mechanism MUST be as good as the legally accepted current state of practice for integrity and authenticity protection).

[491] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms).

[492] Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional)).

7.10. Relation between technical requirements for an increased probative value and the German description of signature functionalities

In Sec. 4.5.1 this thesis listed the functionalities that an electronic signature was intended to fulfil in Germany when the German legislation SigG was released to implement Directive 1999/93/EC following the explanations given in [147, 149]. Those seven functionalities, namely given in BT-Drs 14-4987, have not been added as their own requirement, but have been subsumed in the technical requirements for an increased probative value. If there is a clear relation that a functionality is catered for by a technical requirement this has been marked by a '√' in Tab. 6. Each requirement offered a short pointer or discussion if the connection can be clearly drawn. No mark in Tab. 6 does not exclude that the requirement aids to fulfil or indirectly incorporates the functionality. More details on the functionalities and especially if there are severe differences between classical and malleable signature schemes are given in the concluding Chapter 17.

	Signature functionalities (Sec. 4.5.1)						
Tech. requirements for an increased probative value	Conclusory function	Archiving or integrity function	Identity function	Authenticity function	Verification function	Evidential function	Warning function
Requirement 1 [493]	√	√	√	√	√	√	√
Requirement 2 [494]	√						
Requirement 3.1 [495]		√			√		
Requirement 3.2 [496]			√		√		
Requirement 3.3 (rec.) [497]				√	√		
Requirement 4.1 [498]			√	√			
Requirement 4.2 (opt.) [499]			√	√			
Requirement 5 [500]			√				
Requirement 6 [501]			√	√			√
Requirement 7.1-3 [502]		√	√	√	√	√	
Requirement 8 (opt.) [503]						√	

Table 6. Matrix indicating a clear link between a signature functionality from the official interpretation of the German signature legislation from [149] and [147] and the technical requirements elicited in this thesis for an increased probative value; *rec.=recommended; opt.=optional*

[493] Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer).

[494] Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier).

[495] Requirement 3.1 (Mechanism MUST verify the absence of any subsequent modifications, i.e., ≥1CD integrity).

[496] Requirement 3.2 (Mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory).

[497] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity).

[498] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer).

[499] Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer).

[500] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate).

[501] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)).

[502] Requirement 7.1 (Mechanism MUST achieve cryptographic protection against a forgery),
Requirement 7.2 (Mechanism MUST be as good as the legally accepted current state of practice for integrity and authenticity protection),
Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms).

[503] Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional)).

PART II

Technical requirements for a legally sufficient confidentiality protection for personal data and trade secrets

The biggest threats to our privacy in a digital world come not from what we keep secret but from what we reveal willingly [...].[...].
Restricting these invasions of privacy is a challenge, but it isn't a job for encryption.
Encryption can't protect you from the misuse of data you surrendered willingly.

— Stuart A. Baker
National Security Agency's top lawyer
in WIRED Magazine from June 1994 [21]

Contents

Chapter 8 Confidentiality protection by data removal to protect trade secrets and personal data

Chapter 9 Analysis of existing technical confidentiality and cryptographic privacy definitions

Chapter 10 Technical requirements for legally compliant privacy protection through sanitization

8 — Confidentiality protection by data removal to protect trade secrets and personal data

Overview of Chapter 8

The reasons and uses for authorized modifications with a high probative value are manyfold, e.g., redactions due to the Freedom of Information Act as in the response depicted in Fig. 27 in the motivation. Those applications, as well as those described in the existing literature, show that the reason for a redaction is obvious to hide the original content by redacting it. But also a sanitization shall hide the original, and by this “[...] accommodate different level of data de-identification, supporting the "minimum necessary" disclosure standard of HIPAA Privacy Rule. This provides flexibility not available in redactable signatures.” [12].

In the light of the above this chapter solely focuses on the confidentiality protection needed to successfully protect the confidentiality by removing the following two types of data:

- personal information,
- trade secrets.

Note that the notion of personal data — especially in the non-european context — is also referred to as personally identifying information (PII). In this section the thesis will define the term personal data (see Definition 125 in Sec. 8.2.3) used thereafter as a synonym to PII.

To answer the research question of this thesis (see Sec. 1.3) the focus of Part II and this chapter is to identify legally induced requirements regarding the technical removal of blocks of message to safeguard the confidentiality of the information contained therein. Moreover legal texts are analysed to identify the legal usage of removing information from authenticity-protected content. In the spirit of the quote used on the cover of Part II by Stuart Baker [21][504] the protection is to restrict already the flow of information, not to encrypt it. MSS offer to minimise the information by removing data from a signed data set. Hence, they relinquish the need to share the whole data set just to keep authenticity verifiable.

The chapter concludes with the results of the analysis of several legal texts; an overview of which is given in Fig. 40. The analysis results help to set the technical requirements (see Chapter 10) that mechanisms, which allow to remove information by subsequent modifications, must fulfil to be considered a legally accepted safeguard for sensitive information.

[504] Note, the quote was also quoted by *Steinfeld et al.* in their paper on RSS [455].

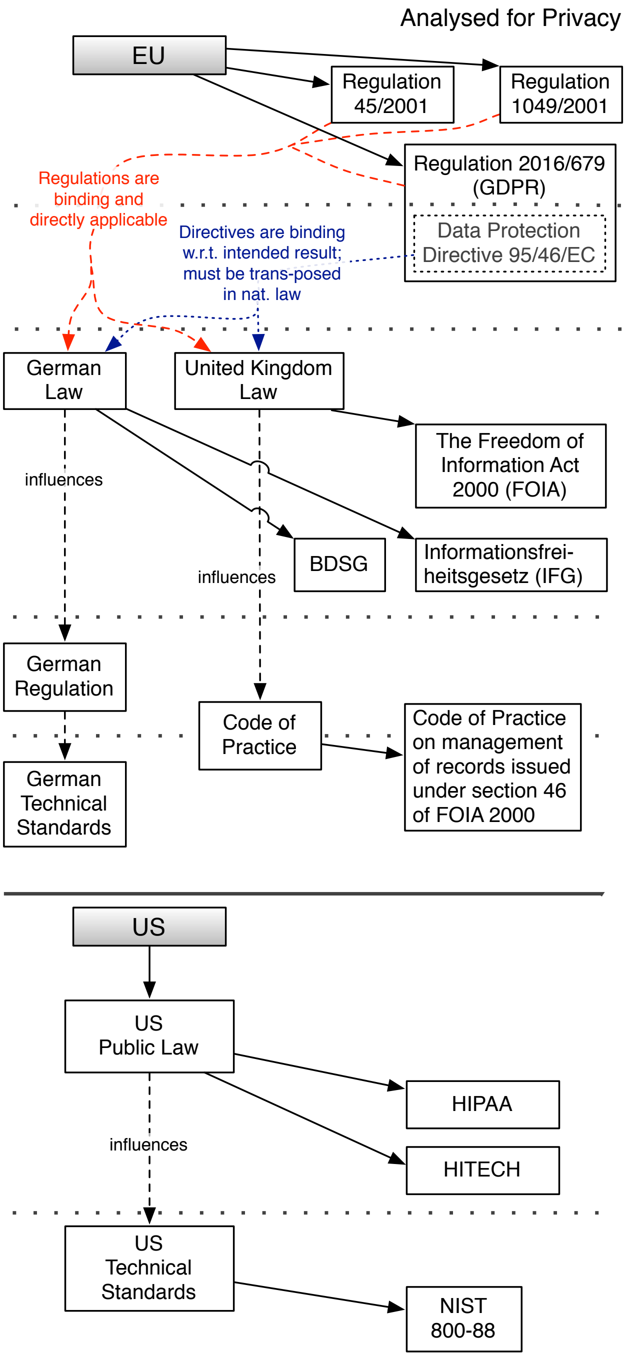

Analysed for Privacy
EU
Regulation 45/2001
Regulation 1049/2001
Regulation 2016/679 (GDPR)
Data Protection Directive 95/46/EC
Regulations are binding and directly applicable
Directives are binding w.r.t. intended result; must be trans-posed in nat. law
German Law
United Kingdom Law
The Freedom of Information Act 2000 (FOIA)
influences
BDSG
Informationsfrei-heitsgesetz (IFG)
influences
German Regulation
Code of Practice
Code of Practice on management of records issued under section 46 of FOIA 2000
German Technical Standards
US
US Public Law
HIPAA
HITECH
influences
US Technical Standards
NIST 800-88

Figure 40. Overview of legal texts analysed in Chapter 8; dashed arrows show influences

8.1. Legal permission to protect confidentiality by the sanitization of information in UK and Germany

The example from Sec. 4.2.2 is a governmental document containing information (see Fig. 27 on page 76) that had to be de-classified before public release[505]. Reasons why data can be redacted can be found in legal texts like:

- German Federal Act Governing Access to Information held by the Federal Government [IFG][506],
- The Freedom of Information Act 2000 from UK [367],

[505] See also example in Sec. 1.2.3 and the statements made by the German Federal Administrative Court (BVerwG) in the ruling from 23.06.2011 - Az. 20 F 21/10, Randnummer 22 [101].

[506] In German 'Gesetz zur Regelung des Zugangs zu Informationen des Bundes (Informationsfreiheitsgesetz - IFG)'

- or the US Freedom of Information Act (FOIA) [490].

The general idea of all above acts is to provide access to information held by governments, but there are circumstances under which the information shall not be given. Then information needs to be withheld, e.g. by redaction as in Fig. 27 that shows a redacted example document given as an answer to a request under the US FOIA.

This section will focus on European legislative texts in the following. This thesis will give the analysis in this section based on the UK's The Freedom of Information Act 2000, in short UK FOIA, as given in [367]. This choice was made to keep the focus of the thesis on the European legislative regime and due to the language being already English and thus forgoing the need to work on a translation. Pointers to German legislative texts are additionally provided as the German legislative regime was heavily used in the analysis of Part I, but only as footnotes.

8.1.1. UK FOIA and German IFG: Reasons for redactions in general

In UK's The Freedom of Information Act 2000 the reasons for withholding information are called "exemptions" [Part II, Sections 21 to 44 UK FOIA][507].

As an example of a reason for an exemption from information release section 24 lists safeguarding national security[508]. There are guidelines on how to interpret the UK act compiled by the he UK's Information Commissioner's Office (ICO), which is an "independent authority set up to uphold information rights in the public interest, promoting openness by public bodies and data privacy for individuals"[509], e.g., a guide dedicated to UK FOIA's Section 24 [479]. The act was followed by a code of practice [482], which clarifies that if information from documents must be withheld due to an exemption, this does not disqualify other information from that document to be released.[510] In detail the code of practice states what could happen if a document contains information which shall not be revealed:

> "Public records to be transferred as subject to an exemption - general
> [...]
> 18.5 Authorities should consider whether parts of records might be released if the sensitive information were **redacted, i.e. rendered invisible or blanked out**. [...]"[511] [482, Para 18.5]

Grammatically interpreted the above suggestion is limited in its scope: it only applies to public records that are transferred to record offices. However, it has been interpreted as "[...] a general reminder of one of the basic features of the Act, namely the right of access is to information not records or documents." [472, p.3].

The UK Freedom of Information Act and the German IFG restrict the release of information, but guidelines say that parts of it might be allowed if sensitive information is redacted. Both legal texts describe under which exemptions information is not released to the public, but kept confidential. However, the quite extensive guidelines released from official UK bodies describe that the confidentiality requirement of sensitive information can be kept by the removal of information from documents by redaction. The Code of Practice on Records Management, issued under Section 46 UK FOIA, states that where a complete document cannot be made available on transfer "Authorities should consider whether parts of records might be released if the sensitive information were redacted [...]" [482, Para 18.5].

This thesis seconds this interpretation of the general idea that the correct interpretation to withhold some information shall not impact the release of the document in general. For example, if future governmental documents that might become subject to FOIA requests would get signed by an MSS then the redactions on the electronic documents keep the authenticity. Of course, the technical mechanisms of

[507] For Germany §§ 3 - 6 IFG list the exemptions and additional guidance is offered in Anwendungshinweise zum Informationsfreiheitsgesetz [82].

[508] "[...] (2) Information which does not fall within section 23(1) is exempt information if exemption from section 1(1)(b) is required for the purposes of safeguarding national security." [Section 24 UK FOIA]

[509] From the website: `https://ico.org.uk` [last accessed: Jun. 2016].

[510] Same holds in Germany; compare the court order from the superior Hessian administrative court (Hessian VGH) in Germany from 02.08.2012 - Az. 27 F 96/11 [228].

[511] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

MSS cannot prohibit the creation of completely — or nearly fully — redacted and thus meaningless governmental documents, like the one depicted in Fig. 27 on page 76 or the one obtained by ACLU [489] which made some news online as it contained “15 pages that were entirely redacted - shaded over in heavy black from top to bottom”[a].

[a] See `http://abcnews.go.com/blogs/politics/2013/05/justice-department-complies-with-foia-by-releasing-completely-redacted-document/` [last accessed: Feb. 2017].

8.1.2. UK FOIA and German IFG: Notion of redaction and its goals

An official guideline emerged following the UK FOIA [482], henceforth referred to as UK FOIA Code of Practice[512]. It was issued following section 46 of the UK Freedom of Information Act 2000. This influence is also depicted in Fig. 40. Additionally, the UK FOIA has been surrounded with guidelines: There is guidance by the Information Commissioner’s Office (ICO) [480] or from the UK’s The National Archives [472]. This section is looking for texts that describe the notion of ’redaction’ and its goals.

The UK FOIA Code of Practice also points[513] to further guidelines: One is a guideline on how to redact and remove information from documents issued by UK’s The National Archives given in [472]. A second one is the guidance documents published by the ICO in [480]. The first of the mentioned guideline provides the following reason for the need of redactions: “[...] one of the basic features of the [Freedom of Information] Act, namely the right of access is to information not records or documents.” [472] The UK’s guidelines by the ICO in [480] and also the ones by The National Archives given in [472] explain the term redaction. They also explain the goal that redaction shall achieve. It is as follows:

> “11. Redacting personal data from the information requested means that **some information** can be released **without** breaching the data protection principles.”[514] [480, p. 4]

From this description by the ICO the thesis infers that the redacted information does no longer contain the “personal data” [480] such that it “can be released without breaching the data protection principles” [480]. Note, in the ICO’s guidance there are additional examples on how to technically correctly[515] or incorrectly[516] carry out redactions [480].

Further, the first of the above mentioned guides — the ‘Redaction Toolkit’ by the national archives [472] — explains the functionality of redaction as follows:

> “What is redaction?
> 3.1 Redaction is the separation of disclosable from non-disclosable information by **blocking out** individual words, sentences or paragraphs or the **removal** of whole pages or sections prior to the release of the document. [...]”[514] [472, p. 4]

This thesis informal functional description of a valid redaction as stated in Definition 46 given in Sec. 3.4.2 is in line with both guides ([472] and [480]). This is emphasised by the explicit note number 4:

> “*In general, a redaction denotes the act of deleting, i.e., removing, one or more blocks of a given message. The redacted block(s) are* ***not reconstructable*** *from the redacted message.*
> [...]
> *Note 4 Definition 46 explicitly states that a redacted block is removed* ***without the possibility to reconstruct it from the modified message****, i.e., its* ***confidentiality is protected*** *as it is irreversibly removed/deleted/hidden.*”[514] [Definition 46]

In the same way, a similar explicit requirement for confidentiality has been codified in this thesis’s Definition 47 for sanitization (see Sec. 3.4.3).

[512] The full title is ’Code of Practice on management of records issued under section 46 of FOIA 2000’.

[513] Page 30 in [482].

[514] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[515] Correctly redacted example `https://ico.org.uk/media/1432972/IneffectiveRedactionExampleEffectiveRedaction.pdf` [last accessed: Aug. 2017].

[516] Not correctly redacted example `https://ico.org.uk/media/1432971/IneffectiveRedactionExampleIneffectiveRedaction.pdf` [last accessed: Aug. 2017].

The UK Freedom of Information Act induced a Code of Practice which describes the confidentiality requirement for the removal of information from documents by redaction. This thesis has put an explicit confidentiality requirement for the original content of subsequently modified blocks in the definition of the redaction and sanitization notions (see Definition 46 and esp. its Note 4). The requirement has been described — not directly termed 'confidentially' though — in natural language in the Code of Practice and can also be found in guidelines, i.e. the UK national archive's 'Redaction Toolkit' [472], that have been released following the UK FIOA Code of Practice [482, p. 30] to guide organisations on how to comply with the law. The following descriptions have been found to be describing the requirement for confidentiality protection by redaction:

- "5. It is therefore essential that any redaction technique is secured to **eliminate the possibility of redacted information being recovered**."[a] [472, p. 15]
- "6.2 Redaction must **irreversibly remove the required information** from the redacted copy of the record. The information must be **completely removed** from the bit stream, not simply from the displayable record."[516] [472, p. 15]
- "[...]redacted, i.e. rendered invisible or blanked out."[516] [482, Para 18.5]

[a] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

8.1.3. UK FOIA and German IFG: Personal data and trade secret related reasons for redactions

This thesis is interested if reasons less drastic than national security[517] are stated for the withholding of data by removal, i.e., can redactions be used to protect personal data or trade secrets? The UK FOIA [367] indeed suggests redactions to remove both of them if necessary.

8.1.3.1. Personal data related exemptions

The exemptions found in Sections 21 till 44 give several times hints to withhold information due to to its affecting person's privacy, e.g., disclosure of court records. Namely, see Part II Sections 21 to 44 UK FOIA. In section 40 data protection legislation is mentioned explicitly as follows:

> "[section] 40.—(1) Any information to which a request for information relates is exempt information if it constitutes personal data of which the applicant is the data subject.
> [...]
> (3) The first condition is–
> (a) in a case where the information falls within any of paragraphs (a)
> to (d) of the definition of "data" in section 1(1) of the **Data
> Protection Act 1998**, that the disclosure of the information to a
> member of the public otherwise than under this Act would contravene–"[516] [Section 40 UK FOIA]

Hence, personal data is clearly marked as an exemption in UK FOIA, making it possible to protect this information by redaction of data. The same is true for the German legal text: § 5 IFG points to German data protection law (BDSG) to clarify when an exemption would be justified[518].

UK FOIA and German IFG point to data protection regulation to clarify possible reasons for exemption. In § 5 (1) IFG, which has the title "Schutz personenbezogener Daten" [92], the German legislation points to the BDSG. Section 40 (3)a UK FOIA points to UK's Data Protection Act 1998. [367]

Once the decision to remove the personal data from the request, it must adhere to the confidentiality requirements, as described above in Sec. 8.1.2.

This thesis will turn to those and other data protection legal texts in Sec. 8.2 in order to find further information on their technical confidentiality protection requirements.

[517] UK FOIA list in section 24 an exemption if withholding information is "required for the purpose of safeguarding national security" [Section 24 UK FOIA].

[518] "[...] Besondere Arten personenbezogener Daten im Sinne des § 3 Abs. 9 des Bundesdatenschutzgesetzes [BDSG] dürfen nur übermittelt werden, wenn der Dritte ausdrücklich eingewilligt hat. [...]" [§ 5 IFG].

8.1.3.2. Trade secret related exemptions

The term trade secret is not further defined in the two legal texts, neither in IFG nor in UK FOIA. The legal definition of the term trade secret is extracted in the following Sec. 8.3. Trade secrets are clearly marked as an exemption in UK FOIA:

> "[section] 43.—(1) Information is exempt information if it constitutes a **trade secret**.
>
> (2) Information is exempt information if its disclosure under this Act would, or would be likely to, prejudice the commercial interests of any person (including the public authority holding it).
> [...]"[519] [Section 43 UK FOIA]

In the above from UK's FOIA, the person can as well be a public authority, a company, or a government department. [481, p. 5] Hence, it can relate to more than one natural person (as defined in Definition 27).

UK FOIA and German IFG explicitly mention protection of trade secrets as reasons for exemption to release the information to the public. § 6 IFG already mentions trade secret (in German 'Betriebs- oder Geschäftsgeheimnis') in the section's title[a] [92]. The UK FOIA lists "commercial interests" [Section 43 (2) UK FOIA] as well as "trade secret" [Section 43 (1) UK FOIA].

If the information is regarded as being needed to be withhold, it must be removed adhering to the confidentiality requirements, as described above in Sec. 8.1.2.

[a] § 6 IFG has the title "Schutz des geistigen Eigentums und von Betriebs- oder Geschäftsgeheimnissen" [92].

8.1.4. EU Regulation 1049/2001: Personal data and trade secret related reasons for withholding information

Additional to member state legislation, also EU legal texts are concerned with the information release. The European Regulation (EC) 1049/2001 "define[s] the principles, conditions and limits on grounds of public or private interest governing the right of access to European Parliament, Council and Commission [...] documents provided [...] in such a way as to ensure the widest possible access to documents" [177, Art. 1]. It contains some exceptions, when documents must not be released to the public in Article 4. Among them are "[...] commercial interests of a natural or legal person, including intellectual property [...]" [177, Art. 4 (2)] as well as "[...] (b) privacy and the integrity of the individual, in particular in accordance with Community legislation regarding the protection of personal data." [177, Art. 4 (1)(b)]. Thus, Regulation (EC) 1049/2001 specifies that information containing personal data or trade secrets shall be withheld and not given to a third party. Moreover, Regulation (EC) 1049/2001 further requests that partial data is to be released by stating that "[i]f only parts of the requested document are covered by any of the exceptions, the remaining parts of the document shall be released." [177, Art. 4 (6)]

EU Regulation 1049/2001 explicitly mentions that for protection of trade secrets or personal data only parts of data not containing them can be released. Article 4 Regulation (EC) 1049/2001 lists a description close to trade secrets[a] and personal data among the reasons for exemptions to the release of data. It does request that if the information to be withheld is only concerning parts of the document, that the "remaining parts of the document shall be released." [177, Art. 4 (6)].

Regulation 1049/2001 does not mandate how information is removed nor that a verifiable authenticity must be kept.

[a] "commercial interests of a natural or legal person, including intellectual property" [177, Art. 4 (2)].

[519] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

8.2. Analysis of confidentiality requirements in definitions of current (personal) data protection law in US, EU, Germany

The analysis in Sec. 8.1.3.1 identified that information that contains personal data can be redacted. In this section the thesis will analyse if the legal texts hold further information about the technical confidentiality protection required for personal information. As a first step several legal definition(s) of the notion personal data contain any guidance about its protection requirements. Note, to answer the research question it is not necessary to define the history[520] nor the notion of privacy, nor what information is legally considered personal data. The following legal texts where selected for an analysis concerning their statements to the confidentiality protection level: Definitions by *Solove*, US HIPAA, US HITECH, US ARRA, EU Regulation 2016/679 (GDPR), BDSG, OECD's privacy guidelines, EU Directive 45/2001 and EU Regulation 1049/2001.

8.2.1. US law: Privacy definition by *Solove*

The following is based on an article and a book by *Solove* [447, 448], which have been used as introductory sources for the notion of privacy in US legislation. Note, the author additionally discusses what information can be regarded as personal data, the interested reader is directed to this work, especially as this is out of the scope[521] of this thesis. The author lists "related problems under the rubric of "privacy"" [448, p. 772], among them are two kinds of "Information Dissemination": "Breach of Confidentiality" and "Disclosure" [448, p. 758]. This harmful activity requires that personal data has already been collected previously [447].

***Solove*'s taxonomy of privacy threats lists loss of confidentiality.** While it is only one of the many aspects that *Solove* lists, the failure to protect the confidentiality of information is an obvious threat. *Solove* further splits it into the "Breach of confidentiality" and "Disclosure" [447]. "Breach of confidentiality is breaking a promise to keep a person's information confidential." [447]. "'Disclosure' occurs when certain true information about a person is revealed to others. Disclosure differs from breach of confidentiality because the harm in disclosure involves the damage to reputation caused by the dissemination; the harm with breach of confidentiality is the violation of trust in the relationship." [447].

Solove does not discuss, at least not in [447, 448], further details on the level of confidentiality protection that his notion of privacy requires. However, he offers some pointers to laws that might discuss this, e.g., The Health Insurance Portability and Accountability Act (HIPAA) of 1996 [447, p. 517]. This law is analysed in the following.

8.2.2. US law: Confidentiality requirements from US HIPAA, HITECH, and ARRA

The US Health Insurance Portability And Accountability (HIPAA) act was established in 1996. Following this, the US Department of Health & Human Services (HHS) published a security standard that is often referred to as the *HIPAA Privacy Rule* and the *HIPAA Security Rule*. The Privacy Rule is located at 45 Code of Federal Regulations (CFR) Part 160 and Subparts A and E of Part 164. There "Confidentiality" is defined in § 164.304 as follows:

Definition 118 : Confidentiality from US HIPAA Privacy Rule

> *"Confidentiality means the property that data or information is not made available or disclosed to unauthorized persons or processes." [486, § 164.304 on p. 8376]*

In the above text from the CFR [486] the notion of redaction cannot be found, let alone a definition. One finds the notion of redaction being used in secondary guidance documentation for HIPAA[522].

[520] Legally the prevailing opinion is that the notion privacy started to shape after *Warren and Brandeis*'s article The Right to Privacy [500].

[521] See statement in Sec. 1.4.3.

[522] "[...] redact the appropriate fields [...]" [491].

In 2009 the Health Information Technology for Economic and Clinical Health (HITECH) Act [485] was enacted. HITECH contains further rules on how to strengthen the privacy and security protections for health information established under HIPAA. Further legal texts from US administrative law explain the goals of confidentiality of health information using the following:

Definition 119 : Confidentiality for personal health information from US law

> *Confidentiality has the goal of "[...] rendering health information unusable, unreadable, or indecipherable."[523] [U.S.R. Title 45 - § 300jj-11 (3) (A) (iv)]*
>
> *Confidentiality can be achieved using "[t]echnologies that allow individually identifiable health information to be rendered unusable, unreadable, or indecipherable to unauthorized individuals when such information is transmitted in the nationwide health information network or physically transported outside of the secured, physical perimeter of a health care provider, health plan, or health care clearinghouse." [U.S.R. Title 45 - § 300jj-12 (2) (B) (vi)]*

With technologies to render "[...] information to be [...] unusable, unreadable, or indecipherable to unauthorized individuals [...]"[U.S.R. Title 45 - § 300jj-12] the US legal text refers to methods using encryption, but also to sanitization of electronic media. Namely, some texts directly refer to NIST SP 800-88 [487][524]. The same phrase for the description of the protection of confidentiality at rest and at transport can be found also in the US ARRA [437] in relation to personal health information (PHI)[525].

US HIPAA and HITECH describe confidentiality goals that require encryption for data at rest and in transit; ARRA uses the term 'redaction'; guidance documents point to technical standards for physical media sanitization. The HIPAA/HITECH acts contain high-level descriptions of the security goal confidentiality, namely "[...] that data or information is not made available or disclosed [...]" [486, § 164.304 on p. 8376] by means of being "[...] rendered unusable, unreadable, or indecipherable [...]" [485, H. R. 1—117]. The American Recovery and Reinvestment Act of 2009 (ARRA) uses the term 'redact' when stating that "[a]ny portion of a report or audit under this subsection may be redacted when made publicly available, if that portion would disclose information that is not subject to disclosure under section 552 of title 5, United States Code (commonly known as the Freedom of Information Act)." [437]. Finally, in related documents released as guidance by US governmental bodies, e.g. [487] giving guidance on HIPAA from the U.S. Department of Health & Human Services, the notions of sanitization can be found and [487] also mention related technical standards, i.e. NIST SP 800-88 for media sanitization, which is thus scheduled for analysis of technical definitions and gets discussed in Sec. 9.3.4.

8.2.3. EU law: Definition of personal data following GDPR (EU Regulation 2016/679)

During the course of this thesis the EU made steps that show that personal data protection is an important goal within the EU. Most importantly, it released an update to the EU legal text of Directive 95/46/EC, which used to be "the reference text, at European level, on the protection of personal data" [Directive 95/46/EC]: Regulation 2016/679 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC, or in short the *General Data Protection Regulation* (GDPR). The latter is a regulation, thus it will become directly applicable in all EU member states. GDPR was adopted in April 2016. It will become legally binding in all member states on 25 May 2018 after a two-year transition period. This thesis notes that there are several legal analyses of the effect of the GDPR on member state regulation and in general, e.g., *Roßnagel and Nebel* state that the GDPR still requires "Member States, supervisory authorities, associations,

[523] The American English spelling from original work is retained for terms and quotes.

[524] "1. Electronic PHI [Protected health information] has been encrypted [...] 2.The media on which the PHI is stored or recorded has been destroyed in one of the following ways: [...] Electronic media have been cleared, purged, or destroyed consistent with NIST Special Publication 800-88, Guidelines for Media Sanitization such that the PHI cannot be retrieved." [487].

[525] "Technologies that allow individually identifiable health information to be rendered unusable, unreadable, or indecipherable to unauthorized individuals when such information is transmitted in the nationwide health information network or physically transported outside of the secured, physical perimeter of a health care provider, health plan, or health care clearinghouse." [437].

the European data protection board and the European legislator need to establish additional legally certain and risk adequate regulations as soon as possible [...]"[526] [416] and *Hornung* notes that "[...] the draft contains many welcome regulations, but also several issues which deserve criticism." [235]. The new rules of the GDPR will become applicable and probably time will make details clearer and show the need or potential for interpretation; a situation comparable with the one after the updated signature regulation where *Roßnagel* noted that an assessment of the regulation's impact will probably only be possible after some court decisions [412, p. 3689].

Regarding the scope of this thesis, it is interested in the required technical protection needs to achieve protection under GDPR and other data protection regulation, not in the question which information is to be protected under GDPR. This thesis is not focussed on defining how and what information is actually identified as being in need of data protection and what legal consequences occur after a breach. This thesis focuses on identifying if MSS can be used as mechanisms for protection. It thus assumes that information gets classified as needing protection and that once this classification is done, the protection of the confidentiality of this information is necessary. Hence, the level of technical protection, i.e., the necessary extent of protection, is of interest to select or tailor the technical mechanisms.

The GDPR states:

> "'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person;" [Art. 4 (1) GDPR].

The GDPR is very recent compared to other data protection related texts ; this thesis defines the term personal data based on the GDPR. Note, in many legal texts the data containing information that requires additional protection under data protection legislation is called *personal data* or *personally identifiable information (PII)*; this thesis sees the two terms as synonyms. Henceforth, the thesis tries to consistently refer to the data covered by GDPR and other data protection or privacy legal texts by the notion 'personal data'. For more details on the definition of the term personal data based on the GDPR see the Analysis Result 14 in Sec. 8.4.1.

The notion of confidentiality is used throughout the GDPR when it comes to describe the risks against which personal data is to be protected against. The analysis found only the following two texts describing what confidentiality is:

> "[...] appropriate security of the personal data, including **protection against unauthorized or unlawful processing** and against accidental loss, destruction or damage, using appropriate technical or organisational measures ('integrity and confidentiality')."[527] [Art. 5 (f) GDPR]
> "[...] confidentiality of the personal data, including for **preventing unauthorized access to or use of** personal data and the equipment used for the processing."[527] [Recital 39 GDPR]

No pointers to technical standards or alike have been found in the text of the GDPR.

The GDPR does use, but not technically define, the term confidentiality or its strength. The most technical description of the intended goal of confidentiality protection is to protection against unauthorized or unlawful processing, access or use. Regarding the strength the statement is that the confidentiality protection shall be "appropriate" [Art. 5 (f) GDPR]. When discussing compliance, the GDPRstates that with respect to the measures to implement "[...] should ensure an appropriate level of security, including confidentiality, taking into account the state of the art and the costs of implementation in relation to the risks and the nature of the personal data to be protected." [Recital 83 GDPR].

[526] The original article [416] is in German; the statement is cited from the english press release from 11.05.2016 found at `http://www.isi.fraunhofer.de/isi-en/service/presseinfos/2016/press-release-14-2016-policy-paper-general-data-protection-regulation.php` [last accessed: Jan. 2017].

[527] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

8.2.4. EU law: 'right to be forgotten' and 'erasure' from EU GDPR

When talking about the confidentiality for personal data, then one 'landmark' clause of the EU GDPR cannot be left out: On 13 May 2014, the Court of Justice of the European Union issued a ruling on the 'right to be forgotten', in relation to online search engines [168]. The GDPR gives several rights[528] to the data subject, among it is the 'right to erasure' in Article 17, which is known as the right to be forgotten. It is mentioned as a phrase in Recital 65 and 66 of GDPR. Regarding the confidentiality required the data shall become "[...] erased and no longer processed [...]" [Recital 65 GDPR] and thus it requires "[...] from the controller the erasure of personal data concerning him or her without undue delay [...]" [Art. 17 GDPR].

The notion and goals of 'erasure' are not further defined in GDPR. While the GDPR mentions that data must be deleted, within the GDPR the notion of 'erasure' is not defined. Furthermore, this thesis found no pointers to other legislation or standards with respect to 'erasure'.[a]

[a] This observation is in line with the findings of [290].

In the next section this thesis discusses the German BDSG's description of the goals of 'erasure'.

8.2.5. German law: 'erasure' from BDSG

As previously found, the GDPR lacks a definition for the term 'erasure'; the German BDSG offers a definition describing the goals of 'erasure' by using the notion of "Unkenntlichmachen" [§ 3 (4) Nr. 5 BDSG] that can be translated as 'to render unrecognizable'[529] In detail, the translation of the German 'Löschung' is 'erasure'[530]. In detail — for the German speaking readers — BDSG states: "Im Einzelnen ist, ungeachtet der dabei angewendeten Verfahren [...] [,] Löschen das Unkenntlichmachen gespeicherter personenbezogener Daten" [§ 3 (4) Nr. 5 BDSG]. Unfortunately, the translation provided[531] by the Language Service of the Federal Ministry of the Interior [91] lost a bit of details when it became translated as follows:

> ""erasure" means the deletion of stored personal data." [§ 3 (4) Nr. 5 BDSG]

To uphold the difference, this thesis suggests to use a term different to the term "deletion" [§ 3 (4) Nr. 5 BDSG] in order to better describes the consequences that are given in the German text:

German BDSG defines the goal of erasure of data as rendering the stored data unrecognisable. This thesis notes that 'unkenntlich' from § 3 (4) Nr. 5 BDSG can be translated with 'unrecognisable' [152][a]. Thus, this thesis suggests that BDSG§ 3 (4) Nr. 5 should be translated as follows to capture the original goals and the intended result of an 'erasure':

> "erasure" means rendering the stored personal data unrecognisable.

This translation is still in line with German legal discussions[b] and governmental guides[c] that also state that erased data must be destroyed irreversibly [138, 440].

[a] Same translation as Langenscheidt `http://de.langenscheidt.com/deutsch-englisch/unkenntlich` [last accessed: Jan. 2017] or it can be translated with 'irrecognisable' according to `https://dict.leo.org/englisch-deutsch/irrecognisable` [last accessed: Jan. 2017].

[b] For example Ulrich Dammann notes in *Simitis*'s Kommentar zum BDSG: "[...] "Unkenntlichmachen" trifft auf jede Handlung zu, die irreversibel bewirkt, dass eine Information nicht länger aus gespeicherten Daten gewonnen werden kann." [440, Randnummer 174–183] .

[c] Compare "Grundgedanke [...] ist, dass durch die Datenlöschung die Kenntnisnahme der auf Datenträgern gespeicherten Daten für jedermann zu jeder Zeit tatsächlich unmöglich ist." [138].

[528] Like the right of information; the deployment of RSS to aid the technical implementation of the right of information is described in Sec. 18.5.

[529] Translated using *Dietl and Lorenz* [152].

[530] Translation according to Langenscheidt Recht Englisch [70] or *Dietl and Lorenz* [152].

[531] At `gesetze-im-internet.de`.

8.2.6. International law: OECD privacy guidelines

On international level the Organisation for Economic Co-operation and Development (OECD) had issued privacy guidelines in 1980. They contain principles also found in other legislative texts today. In 2013 the OECD Council updated them and issued a revised "Recommendation Concerning Guidelines Governing the Protection of Privacy and Transborder Flows of Personal Data ("Privacy Guidelines")" [360]. "These new Guidelines constitute the first update of the original 1980 version that served as the first internationally agreed upon set of privacy principles."[532].

There are several principles, out of which this thesis sees three as indirectly related to the focus of keeping personal data confidential.

Definition 120 : Security Safeguards Principle from OECD

> *"Personal data should be protected by reasonable security safeguards against such risks as loss or unauthorized access, destruction, use, **modification** or disclosure of data."[533] [360]*

In the security safeguards principle one can clearly identify aspects regarding confidentiality protection, e.g. the terms "unauthorized access" [360] or "disclosure of data" [360]. But also of integrity protection, i.e., "modification [...] of data" [360]. All in all the classical goals of confidentiality, integrity and availability are reflected in the principle from Definition 120.

Definition 121 : Individual Participation Principle from OECD

> *"An individual should have the right:*
> *[...]*
> *d) to challenge data relating to him and, if the challenge is successful to have the data **erased**, rectified, completed or amended."[533] [360]*

Within the individual participation principle the OECD encoded the right for erasure, i.e., similar to one found in the EU GDPR as discussed in Sec. 8.2.4. Likewise, the OECD does not specify in [360] what "erasure" technically shall be comprised of.

Definition 122 : Collection Limitation Principle from OECD

> *"There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject." [360]*

In this third principle, which is actually the first of eight mentioned in [360], it becomes clear that personal data that is inappropriate for the process to be known shall never become collected.

Finally, there is a forth principle, that together with the integrity protection requirement from the safeguard principle is of interest with respect to the integrity protection discussed: the data quality principle.

Definition 123 : Data Quality Principle from OECD

> *"Personal data should be relevant to the purposes for which they are to be used, and, to the extent necessary for those purposes, should be accurate, complete and kept up-to-date." [360]*

The data quality principle can — as well as the safeguard principle — be interpreted to suggest that integrity protection for personal data at rest could be beneficial. Of course if the data is inaccurate and does not represent the real world facts then its veracity (see short discussion in Sec. 5.1.8) is not given, which is an other property related to integrity, but not equal to it.

The OECD identifies principles for privacy that include integrity and confidentiality of already collected and limitation of collection of new data. The OECD privacy framework identifies several principles for privacy in which the OECD clearly hints that confidentiality must be protected, while data is needed.

[532] From `http://www.oecd.org/internet/ieconomy/privacy-guidelines.htm`.

[533] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

Additionally, the framework states in the data quality principle (Definition 123) and in the integrity related statement of the security safeguards principle (Definition 120) that it is deemed to be beneficial if data is protected against unauthorized modifications, such that the data stays accurate.

The OECD document itself does not specify the technical requirements for confidentiality in [360].

8.2.7. EU law: Definition of data minimisation principle from EU GDPR

This principle can be found in European and German[534] legal texts, e.g. Directive 95/46/EC, GDPR or BDSG. In detail the Article 5 GDPR on "Principles relating to processing of personal data" describes the data minimisation principle as follows:

> "1. Personal data shall be:
> [...]
> (c) adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed ('data minimisation');" [Art. 5 GDPR]

The German BDSG likewise demands that systems are to "[...] be designed in accordance with the aim of collecting, processing and using as little personal data as possible." [§ 3a BDSG]. Moreover, the German BDSG demands that data minimisation shall be done "[...] as far as possible and the effort involved is reasonable in relation to the desired level of protection." [§ 3a BDSG].

One of the latest European legal texts on personal data protection — the GDPR — defines the principle as follows:

Definition 124 : Data Minimisation Principle following EU GDPR

> *The principle of data minimisation dictates that the personal data collected must be "[...] adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed" [Art. 5 GDPR].*

In a guidance document from ENISA on the cryptographic protection of personal data [187] the principle of data minimisation has been summarised and motivated as follows:

> "[T]he best protection for personal data is to not have collected the data at all. Data minimization can be seen as an example of the principle "prevention is better than reaction." Here, "prevention" means that one tries to avoid storing personal data as much as possible, thereby reducing the need for "reaction", i.e. securing stored data."[535] [187]

The EU GDPR and the German BDSG seek to minimise data before data processing. Within the EU GDPR one principle is to minimise the amount of personal data that is processed. Note that processing includes the communication and storage of data. In light of the data minimisation principle the removal of unnecessary information from data before storing or communicating it becomes a legal necessity. The German § 3a BDSG additionally contains a clause that is balancing the respective interests of the data processor in terms of overhead.[a] Reducing the amount of personal data helps "reducing the need for [...] securing stored data." [187].

In both regards, having a malleable signature scheme that keeps the original data confidential after sanitization or redaction will be beneficial. First and most obviously, it will allow to remove the original information. Second, an MSS will reduce the effort to remove data from authentic documents while keeping them authentic: It is no longer necessary to enter into an interaction with the original signatory to retrieve a signature for a document that contains all except the undesired information. With conventional signatures this would be the only way to retain a verifiable authenticity on the redacted document. Thus, with an MSS there would be less room for argumentation of the data processor as there is no additional effort to fully sanitize or redact the unnecessary personal data from a signed document if the data processor is authorized to do so.

[a] "In particular, personal data are to be aliased or rendered anonymous as far as possible and the effort involved is reasonable in relation to the desired level of protection." [91].

[534] In German § 3a BDSG calls it "Datenvermeidung und Datensparsamkeit", which can be translated into "Data reduction and data economy" [91].

[535] The American English spelling from original work is retained for terms and quotes.

8.2.8. EU law: Directive 45/2001 and Regulation 1049/2001

The European Regulation (EC) 45/2001 has the purpose to "[...] protect the fundamental rights and freedoms of natural persons, and in particular their right to privacy with respect to the processing of personal data [...]" [176, Art. 1 (1)]. It is targeting the data processing done by EU institutions and bodies. It contains definitions of terms and regulations alike the EU Directive 95/46/EC. The latter got superseded by the GDPR. Following recital 17 of the GDPR the Regulation 45/2001 will be interpreted or even adapted according to the newer GDPR:

> "Regulation (EC) No 45/2001 of the European Parliament and of the Council (2) applies to the processing of personal data by the Union institutions, bodies, offices and agencies. Regulation (EC) No 45/2001 and other Union legal acts applicable to such processing of personal data should be adapted to the principles and rules established in this Regulation and applied in the light of this Regulation. In order to provide a strong and coherent data protection framework in the Union, the necessary adaptations of Regulation (EC) No 45/2001 should follow after the adoption of this Regulation, in order to allow application at the same time as this Regulation." [Recital 17, GDPR]

The EU Regulation 45/2001 applies to data processing by the EU institutions and bodies, but follows the GDPR. Within GDPR's recital 17 the EU stated their intent to interpret or if necessary adapt Regulation 45/2001 such that it is aligned with the principles and rules from GDPR. Note, Regulation 45/2001 neither mentions the terms 'redaction' or 'sanitization' nor defines the level of confidentiality in any detail. Thus Regulation 45/2001 on its own is not further analysed in this thesis.

The European Regulation (EC) 1049/2001 has the purpose "[...] to define the principles, conditions and limits on grounds of public or private interest governing the right of access to European Parliament, Council and Commission [...] documents provided [...] in such a way as to ensure the widest possible access to documents" [177, Art. 1]. It contains some exceptions when documents must not be released to the public in Article 4. Among them are the need to protect national security or if the release would interference with legal or ongoing decision making processes. While it does contain rules on information classified as secret it does not describe any details on the confidentiality protection. However, Regulation 1049/2001 gives rise to the idea of redacting or releasing documents only partially when it states that "[i]f only parts of the requested document are covered by any of the exceptions, the remaining parts of the document shall be released." [177, Art. 4 (6)].

EU Regulation 1049/2001 does not explicitly mention the confidentiality level for the protection of data that has been removed. Article 4 Regulation (EC) 1049/2001 clearly states that if the information to be withheld is only concerning parts of the document, that the "[...] remaining parts of the document shall be released." [177, Art. 4 (6)]. While it also contains an article on "Treatment of sensitive documents" [177, Art. 9] it makes no explicit statement if the removal of parts is possible or what level of confidentiality protection must be offered.

As no statement regarding confidentiality protection is made, this thesis assumes that the information in need to be withheld, must be completely removed, and thus only the remaining parts become accessible to the requestor. Hence, a high confidentiality guarantee is required.

8.3. Legal definition of trade secret protection in UK, EU and international law

The analysis in Sec. 8.1.3.2 identified that information constituting a trade secret can be redacted. As a first step the legal texts are analysed to find if the definition of the notion trade secret holds information about its protection requirements. In legal terminology, information that is kept confidential in order to preserve competitive gains is referred to as 'trade secrets', 'undisclosed information' (see for example TRIPS section 7 Article 39 [44]), 'business confidential information' or 'secret know-how'. Business and academia sometimes use other name tags for it such as 'proprietary know-how' or 'proprietary technology'. This section presents some background on the legal term 'trade secret'. As examples of trade secrets the UK Ministry of Justice lists "secret processes of manufacture" [481] or "special formulae" [481]. In summary this leads to the description what constitutes the legal notion of a trade secret for this thesis in Sec. 8.4.2.

8.3.1. UK law: Legal definition of trade secret

Following the UK Ministry of Justice's guidance "[t]he Freedom of Information Act does not define a trade secret, nor is there a precise definition in English law." [481]. Thus, this thesis did not search further for such a definition. However, the guidance text further states — referring to the case of Lansing Linde Ltd. v Kerr (1991) 1 WLR 251 — that:

> " [...] [the trade secret] must be information used in a trade or business
>
> - it is information which, if disclosed to a competitor, would be liable to cause real (or significant) harm to the owner of the secret
> - the owner must limit the dissemination of the information, or at least, not encourage or permit widespread publication
>
> [...]" [481]

8.3.2. EU law: Legal definition of trade secret

In the legal context of the EU, one finds a definition of the term "trade secret" in Article 2 of the Directive (EU) 2016/943 on the protection of undisclosed know-how and business information (trade secrets) against their unlawful acquisition, use and disclosure [183]. The Directive (EU) 2016/943 Article 2 (1) states the following:

> " 'trade secret' means information which meets all of the following requirements:
>
> (a) it is secret in the sense that it is not, as a body or in the precise configuration and assembly of its components, generally known among or readily accessible to persons within the circles that normally deal with the kind of information in question;
>
> (b) it has commercial value because it is secret;
>
> (c) it has been subject to reasonable steps under the circumstances, by the person lawfully in control of the information, to keep it secret; [...]" [183, Art. 2 (1)]

8.3.3. International law: Legal definition of trade secret from WTO

The term trade secret is closely related to the legal term of intellectual property (IP) and the rights assigned to it[536]. The security protection needed for an IP, that is regarded a financial asset as it is able to attract an investment, is to keep the IP secret, such that it cannot be accessed from unauthorized parties. The protection of intellectual properties is also subject to regulations of the World Trade Organization (WTO), where you find a similar wording in the TRIPS agreement[537] [44, 506]. Article 39 in section 7 of TRIPS states the following:

> " 2. Natural and legal persons shall have the possibility of preventing information lawfully within their control from being disclosed to, acquired by, or used by others without their consent in a manner contrary to honest commercial practices[10] so long as such information:
>
> (a) is secret in the sense that it is not, as a body or in the precise configuration and assembly of its components, generally known among or readily accessible to persons within the circles that normally deal with the kind of information in question;
>
> (b) has commercial value because it is secret; and
>
> (c) has been subject to reasonable steps under the circumstances, by the person lawfully in control of the information, to keep it secret.

[536] So-called intellectual property rights (IPR).

[537] By 'TRIPS' or 'TRIPS agreement' this thesis refers to the agreement on Trade-Related Aspects of Intellectual Property Rights, Annex 1C to the Agreement Establishing the World Trade Organization (WTO).

> 10) For the purpose of this provision, "a manner contrary to honest commercial practices" shall mean at least practices such as breach of contract, breach of confidence and inducement to breach, and includes the acquisition of undisclosed information by third parties who knew, or were grossly negligent in failing to know, that such practices were involved in the acquisition. " [506, article 39 section 7]

The TRIPS wording describes a protection against information disclosure to third parties acting unlawfully and without consent. This comes close to the information security protection goal of confidentiality by ISO 27000 (see Definition 131 given in Sec. 9.2.1): the technical definition from ISO 27000 requires that confidential information is "not made available or disclosed to unauthorized individuals" [254]. Note, of course the general definition of confidentiality from ISO is not limited protect only commercially valuable information. This observation leads to the decision to use WTO's definition of a trade secret for this thesis, see Analysis Result 15 stated in Sec. 8.4.2.

EXAMPLE

As an example, again from the domain of supply chains, note that the participants — like suppliers, producers, and consumers of goods — realise an inter-organisational process. Due to this high interaction and interlinking of processes the safety of the output product is affected by any error in any of the supplied inputs of any previous production steps. Once a defect in an input product is identified it is crucial to identify all the affected output products. If such a defect is only recognised after some time, then traceability solutions, as prescribed by legal texts concerning food safety, aid in identifying the cause and the effects of the defect. However, this poses a problem since a company might consider the names of business partners or customers their trade secret or at least has a legitimate interest in keeping the sources confidential[538]. If a business partner gives sub-tasks to sub-partners and then tries to unrecognisably hide that the sub-partner's work was incorporated into his own. Or the business partner is in the role of a reseller, thus hiding his own suppliers is crucial for his business. In all these situations mechanisms are needed to unrecognisably hide this information from an electronic document, which is authenticity-protected by a trustworthy entity like an accredited testing laboratory, while keeping the authenticity of the remaining information as good as possible. Even if for example the business partner's name is not considered to be secret, additional data about the goods and the terms of trade often are.

Decision what information from a document to withhold and which to release is often taken after the creation of the signature. The problem to decide which information has to be considered *internal* and which may be shared with a partner cannot always be decided when the integrity and authenticity protection is applied at the document's origin, i.e. when its signed. Only later the document holder can identify what information needs to unrecognisably removed from the signed document before sharing. Only then the document holder can ensure that all relevant contextual information that is at hands of the communication partner can be considered to decide what to withhold by a redaction or sanitization.

This observation leads to the formulation of Analysis Result 18 stated in Sec. 8.4.5.

8.3.4. EU law: Personal data versus trade secrets in the Directive 95/46/EC

In May 2018 the EU Directive 95/46/EC will be superseded by the GDPR. Still interestingly, the Directive 95/46/EC [174] balances the right of access[539], which grants any person the right to obtain information about the processing of his or her personal data, with the protection of trade secrets. Namely, it describes that this right has to be exercised such that neither it "adversely affect[s] trade secrets" [Recital 41 Directive 95/46/EC] nor protection of trade secrets results "in refusing all information" [Recital 41 Directive 95/46/EC]. As such it allows the controller to refuse providing the information on data undergoing processing or on the logic involved in any automatic processing. Alternatively, the controller is allowed to reduce the provided information about the recipient of data to an abstract level, i.e., list them only by category instead of name[540] [Article 18 2. Directive 95/46/EC].

[538] Confirmed by OLG Hamburg, 5 U 106/08 of 15.04.2010 [362] in case of partial blackened information about suppliers.

[539] Also the German BDSG knows the "Auskunft an den Betroffenen" [§ 19 BDSG].

[540] See for example "[...] the recipients or categories of recipient to whom the data are to be disclosed [...]" [Art. 18 2. Directive 95/46/EC] or "[...] die Empfänger oder Kategorien von Empfängern, an die die Daten weitergegeben werden [...]" [§ 19 (1) 2. BDSG].

The EU Directive 95/46/EC balances the right of access with the confidentiality protection of trade secrets. A request from the data subject for information on the processing of his or her personal data must not include all information if the release "[...] adversely affect[s] trade secrets [...]" [Recital 41 Directive 95/46/EC]. Of course trade secret protection cannot be used as a blanket excuse to not answer valid requests.

How an MSS can be facilitated in this respect is one of the application of MSS presented in this thesis and can be found in Sec. 18.5.

8.3.5. EU and German law: No general right to trade secret protection

Under certain legal circumstances there is no right to uphold the protection of own trade secrets. One example is from food and feed business related law which is seeking to protect the public health and safety from harmful food products. Article 18 and 19 of EC Regulation 178/2002 [178] are requiring that "[f]ood and feed business operators shall have in place systems and procedures to identify the other businesses to which their products have been supplied. This information shall be made available to the competent authorities on demand." [178, Art. 18 para. 3]. In this respect, there is no mention that one is able to withhold information from the "competent authorities" [178], e.g., the state. However, the food and feed business operators shall be able to assume that unless there is a real need to release that information to the public, the "competent authorities" [178] will keep it confidential and follow stringent procedures before releasing this information, e.g., due to a FIOA request, as discussed in Sec. 8.1.3.2.

Further, the German Verbraucherinformationsgesetz und Offenlegungspflichten (VIG) [150] since September 2012 contains a description that clearly states that food and feed business operators cannot simply use trade secret protection as a reason to withhold certain information[541] and especially not if the public health is endangered[542]. However, the VIG still says that food safety is not interested in information related to recipes[543], which are often considered a trade secret.

EU legal texts exempt confidentiality protection for trade secrets if this information is considered important to protect the public health and safety. In EU regulation related to food and feed, and namely in the German (VIG) [150], the protection of public safety is given priority over trade secret protection.

This finding has added to the analysis result that the right to confidentiality protection for data might vary over time formulated as Sec. 8.4.5.

8.4. Analysis results of legal definitions regarding confidentiality protection of personal data and trade secrets

To conclude, this section compiles in condensed and summarised form the intermediate findings and results obtained from the thorough analysis of the legal texts. The results, together with the analysis of more technical and computer science related texts followed hereafter in Chapter 9, form the basis of this thesis requirements for confidentiality presented in Chapter 10.

[541] "Der Zugang zu folgenden Informationen kann nicht unter Berufung auf das Betriebs- und Geschäftsgeheimnis abgelehnt werden [...]" [150, § 3].

[542] "Berufung auf Betriebs- oder Geschäftsgeheimnisse entfällt
Künftig müssen die amtlichen Kontrollergebnisse der Lebensmittelüberwachung bei allen Messergebnissen, die Grenzwerte, Höchstmengen oder Höchstgehalte betreffen, herausgegeben werden. Eine Berufung auf Betriebs- oder Geschäftsgeheimnisse ist nicht mehr möglich. Dies gilt unabhängig davon, ob die Grenzwerte überschritten worden sind oder nicht. Bei Rechtsverstößen wird zusätzlich klargestellt, dass die komplette Lieferkette offengelegt werden muss. Generell gilt ab jetzt: Ein Geheimnisschutz kommt nicht in Betracht, wenn das öffentliche Interesse an einer Herausgabe der Information überwiegt." [189].

[543] "Klargestellt ist aber jetzt auch im Gesetz: Rezepturen und sonstiges exklusives technisches oder kaufmännisches Wissen bleiben auch weiterhin geschützt." [189].

8.4.1. Analysis Result 14: Definition of personal data

To stay consistent, this thesis chose to use the definition from the GDPR as the definition for personal data. After the analysis of several legal definitions (see Sec. 8.2) this thesis notes that in many legal texts the data containing this information is also named *personally identifiable information (PII)*; this thesis sees this as synonymous to the term *personal data*. Henceforth, the thesis bases on the following notion derived from GDPR:

Definition 125 : Personally identifying information (PII) / personal data from GDPR

> *" 'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person;" [Art. 4 (1) GDPR]*

8.4.2. Analysis Result 15: Definition of trade secret

From the analysis of several legal definitions (see Sec. 8.3) this thesis chose to use the definition from the WTO as the definition for trade secrets. It describes that the information is considered a trade secret mainly because the data has a commercial value due to its secrecy and that its secrecy is protected against third parties.

To prohibit that information is "[...] made available or disclosed to unauthorized individuals [...]"[254] is also the protection goal of confidentiality following technical definitions (like ISO 27000; see Definition 131 given in Sec. 9.2.1).[544] Thus, this legal definition is selected due to being already in line with some technical definition. More technical definitions are analysed in Chapter 9.

Textually, the only change from the original text found in TRIPS (as presented in Sec. 8.3.3) is the removal of numbering and the footnote, and a change in the layout.

Definition 126 : Trade Secret following WTO TRIPS [506]

> *"Natural and legal persons shall have the possibility of preventing information lawfully within their control from being disclosed to, acquired by, or used by others without their consent in a manner contrary to honest commercial practices so long as such information:*
>
> *(a) is secret in the sense that it is not, as a body or in the precise configuration and assembly of its components, generally known among or readily accessible to persons within the circles that normally deal with the kind of information in question;*
>
> *(b) has commercial value because it is secret; and*
>
> *(c) has been subject to reasonable steps under the circumstances, by the person lawfully in control of the information, to keep it secret.*
>
> *For the purpose of this provision, "a manner contrary to honest commercial practices" shall mean at least practices such as breach of contract, breach of confidence and inducement to breach, and includes the acquisition of undisclosed information by third parties who knew, or were grossly negligent in failing to know, that such practices were involved in the acquisition." [506, article 39 section 7]*

[544] Note, the definition of confidentiality from ISO 27000 [254] is more general and is not limited to the protection of only commercially valuable information.

8.4.3. Analysis Result 16: Legal texts acknowledge the importance of confidentiality protection of personal data and the high-level descriptions of the goal of confidentiality coincide

The German BDSG as well as the GDPR have put up the principle of personal data minimisation. The OECD likewise lists data collection limitation as one of its principles. All those principles share that a privacy-conscious system must limit the collection, processing and sharing of personal data to the least possible amount. If data is transmitted from one process to another, collecting information happens by receiving it. Thus, non-transmission of unnecessary information would result in limiting the collection at the receiving process. Hence, if information already exists in a data set and this is considered personal information, but unnecessary for the next task, it must not be forwarded and thus removed from the data set given to other parties for further processing.

In general, the protection of confidentiality of personal data plays an important role in personal data protection laws; *Solove* lists the failure to protect the confidentiality of the information as “Breach of confidentiality” [447] and “Disclosure” [447] as threats to privacy.

Also the guidance documents introduced following the legal acts often define the requirements towards the confidentiality required more precisely, like the Code of Practice [482] introduced following UK FOIA states that information shall be “[...] rendered invisible or blanked out [...]” [482, Para. 18.5]. The German BDSG also defines the goal of erasure of data to be rendering the stored data unrecognisable (see discussion in Sec. 8.2.5) and thus gives a direction for the protection that a mechanism must offer. Table 7’s third column provides an overview.

While the strength or the technical means are not given, the legal texts that have been analysed do not differ much in their functional description of the goal that technically is termed confidentiality. Table 7’s second column shows an overview. The GDPR lists ”[...] preventing unauthorized access to or use of [...] “ [Art. 5 (f) GDPR] as the goal. The German BDSG describes as well that confidentiality requires that one “[...] shall not collect, process or use personal data without authorization [...]” [§ 5 BDSG]. In the US law the analysed texts converge towards the goal that data MUST be “[...] rendered unusable, unreadable, or indecipherable to unauthorized individuals [...]”[545] [U.S.R. Title 45 - § 300jj-12] to be seen as confidentiality protected. Due to this alignment the thesis selected the following statement—from the US legal texts on HIPAA—as a legal definition of confidentiality:

Definition 118 : Confidentiality from US HIPAA Privacy Rule

> *“Confidentiality means the property that data or information is not made available or disclosed to unauthorized persons or processes.” [486, § 164.304 on p. 8376]*

Just for the purpose of readability the US legal texts are grouped into one line in Tab. 7.

[545] The American English spelling from original work is retained for terms and quotes.

Definition	implicitly describe confidentiality protection	explicitly mention sanitization or redaction
German Federal Act Governing Access to Information held by the Federal Government [IFG]	–	✓ [546]
UK Freedom of Information Act 2000 [367]	(✓) [547]	(✓) [548]
EU Regulation 2016/679 (GDPR)	✓ [549]	–
German BDSG	✓ [550]	–
EU Directive 45/2001	–	–
EU Regulation 1049/2001	✓ [551]	–
OECD's privacy guidelines	✓ [552]	–
Definitions by *Solove*	(✓) [553]	–
US HIPAA, US HITECH, and US ARRA	✓ [554]	✓ [555]

Table 7. Overview of the analysed legal texts and their discussion of the required confidentiality protection as well as their use or definition of terminology related to sanitization or redaction; "–" means no or not discussed; "(✓)" means indirectly

8.4.4. Analysis Result 17: Legal texts demand the confidentiality protection of trade secrets if not outweighed by common interests

The analysis of the UK FOIA and the German IFG yielded that also trade secrets can be confidentiality protected by redactions. § 6 IFG has trade secret in the section's title[557] [92]. The UK FOIA lists "commercial interests" [Section 43 (2) UK FOIA] as an exemption as well as "trade secret" [section 43 (1) UK FOIA]. Also the European Regulation (EC) 1049/2001 lists "[...] commercial interests of a natural or legal person, including intellectual property [...]" [177, Art. 4 (2)] as a reason to exempt that information. However, note that there are limits to the possibility to withhold information on the basis of it being a trade secret, e.g., when the information is requested by governmental bodies, which then have to check before releasing information to the public, or if the public health is endangered, as discussed briefly in Sec. 8.3.5.

[546] See (Hessian VGH) in Germany from 02.08.2012 - Az. 27 F 96/11 [228].

[547] Indirectly in an official guidance document it states "[...] eliminate the possibility of redacted information being recovered." [472, p. 15].

[548] Indirectly in official guidance documents: in [472] and in [482]; the latter states "[...] redacted, i.e. rendered invisible or blanked out." [482, Para 18.5].

[549] "[...] preventing unauthorized access to or use of [...] " [Art. 5 (f) GDPR].

[550] "Persons employed in data processing shall not collect, process or use personal data without authorization (confidentiality)" [§ 5 BDSG][551] translated from the German text stating "Den bei der Datenverarbeitung beschäftigten Personen ist untersagt, personenbezogene Daten unbefugt zu erheben, zu verarbeiten oder zu nutzen (Datengeheimnis)." [§ 5 BDSG].

[552] As only the "[...] remaining parts of the document shall be released" [177, Art. 4 (6)]. this thesis assumes that the information in need to be withhold, must be completely removed, and thus become inaccessible to the requestor.

[553] The Security Safeguards Principle from OECD is formulated using notions of confidentiality protection, i.e. "unauthorized access" [360] or "disclosure of data" [360].

[554] The author lists "Breach of Confidentiality" [448, p. 758] and following US administrative law that quite specifically defines confidentiality to mean that "[...] information [is] to be [...] unusable, unreadable, or indecipherable to unauthorized individuals [...]" [U.S.R. Title 45 - § 300jj-12].

[555] US HIPAA demands "[...] that data or information is not made available or disclosed [...]" [486, § 164.304 on p. 8376] by means of being "[...] rendered unusable, unreadable, or indecipherable [...]" [485, H. R. 1—117] as stated in US HITECH.

[556] US ARRA uses the term 'redact' when stating that "[a]ny portion of a report or audit under this subsection may be redacted when made publicly available, if that portion would disclose information that is not subject to disclosure under section 552 of title 5, United States Code (commonly known as the Freedom of Information Act)." [437]

[557] § 6 IFG has the title "Schutz des geistigen Eigentums und von Betriebs- oder Geschäftsgeheimnissen" [92] and 'Geschäftsgeheimniss' translates into trade secret according to Langenscheidt Recht Englisch [70].

8.4.5. Analysis Result 18: The importance of or the right to confidentiality protection for data might vary over time

Following the discussion from Sec. 8.3.3 this thesis notes that in many applications no prediction of how the data is actually best composed can be made at the time when integrity protection is applied. Many legal circumstances must be considered during the selection of information to be shared as to whether the information is considered a trade secret, personal data, or another class that would grant it an exemption and thus would entitle it to be withhold. The need for confidentiality protection might be arising at a time after signature generation. In the same way data that might be usually be protected for confidentiality because it is considered a trade secret might later be required to be revealed and its authenticity might become challenged, as discussed in Sec. 8.3.5.

EXAMPLE

E-discovery is one example for a process that might require the very special selection of data blocks that become revealed from a larger set while keeping their authenticity [166, 261, 324]. In e-discovery a party involved in a court case is requested to provide to the court all digitally stored information relevant to that case. Of course the party does not want to release too much information, but it shall also not withhold information that is related. Further it might be helpful if the information released and in favour of the party is considered authentic by the court and will have an increased probative value.

EXAMPLE

Another example is the secondary use of medical data "[...] for clinical studies. Therefore, it is absolutely necessary to anonymize medical data by removing patient identifying information or by providing solely parts of the documents which are relevant for a specific study. But this can only be accomplished on demand and depends on the context. This means, that a priori anonymization of medical data is not feasible without knowing the intention of the study and the respective sample. This is for instance due to the fact that k-anonymity [462] should be achieved, which strongly depends on the used sample."[558,559] [444]

A redactable or sanitizable signature scheme is able to protect the confidentiality of data by subsequent redactions or sanitizations and keeps the authenticity of the data as it is protected against unauthorized modifications. To be flexible the analysis yields that the scheme at best

(1) limits the disclosure to as little information as possibly required[560],

(2) postpones the decision of what information is removed (redacted) or modified (sanitized) to a time as late as possible,

(3) requires no involvement of the original document's signatory[561].

Another case in support of this analysis result can be found in the guidances related to UK FOIA. The guidance given in [481] state that the holder of the potential trade secrets could be involved[562] in the process of the decision. However, the same guidance establishes that asking if the party considers the data a trade secret is not always necessary, nor shall their statement bind the administrations' final decision[563]. Hence, the third party who provided the signed data might have no or little[564] influence on what will get redacted.

[558] Original citation was adjusted to point to [462].

[559] The American English spelling from original work is retained for terms and quotes.

[560] Namely, this means that the overhead for each block must be small as more fine-grained redactions or sanitizations might mean large numbers of blocks.

[561] Namely, contacting the originator might be a leak of information in its own right.

[562] "Where a disclosure would be likely to prejudice the interests of a third party, it may be appropriate to consider an **approach** to **that third party** to try to establish its willingness or ability to waive or mitigate that prejudice." [481, p. 7].

[563] "It must be stressed that the prejudice to third party interests has to be assessed objectively and by the authority, [...]" [478, p. 7].

[564] The guidance text establishes that specific contractual confidentiality clauses can help to identify which information gets classified as trade secrets by stating: "During the procurement process public authorities may be asked by contractors to accept confidentiality clauses which attempt to prevent the disclosure of information. In many cases such clauses may be perfectly proper and serve to make clear that information which the supplier considers should not be made public and that which can be freely disclosed." [478, p. 10].

8.4.6. Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism

This thesis focuses on malleable signature schemes as a cryptographic mechanism to achieve confidentiality for the removed or overwritten information. From a functional point of view, the focus lies on the two functionalities of redact and sanitize (see Sec. 3.4.2 and Sec. 3.4.3 for a general introduction to the notions). This view is supported by legal texts, as several mention redactions as a valuable technical mechanism to not disclose some information. The legal texts analysed have the goal to foster the release of information into the public domain, e.g., UK FIOA [367] or German IFG [92], but enlist personal data as well as trade secrets as exemptions for a public release. Namely, the Code of Practice [482], which was introduced following UK FOIA, clarifies that if information from documents must be withheld due to an exemption, this does not disqualify other information from that document to be released. The same principles can be seen in Germany[565]. In detail the Code of Practise [482] states what could happen if a document contains information which shall not be revealed:

> "Public records to be transferred as subject to an exemption - general
> [...]
> 18.5 Authorities should consider whether parts of records might be released if the sensitive information were **redacted, i.e. rendered invisible or blanked out**. [...]"[566] [482, Para 18.5]

Also another UK guidance texts from the UK National Archives [472] explains the technical goals of a redaction as follows:

> "[A]ny redaction technique is secured to **eliminate the possibility of redacted information being recovered**."[566] [472, p. 15]
> "Redaction must **irreversibly remove the required information** from the redacted copy of the record."[566] [472, p. 15]

Finally, the analysis also found the notion of 'sanitization' being mentioned in US legal texts. However, in the texts analysed it was mentioned with regards to the physical storage media of the data to be deleted [487]. Still, the notion was mentioned in the context of "[...] rendering [...] information unusable, unreadable, or indecipherable."[567] [U.S.R. Title 45 - § 300jj-11 (3) (A) (iv)].

This gives no rise to assume any misalignments of the notions 'sanitization' and 'redact'.

[565] Compare the court order from the superior Hessian administrative court (Hessian VGH) in Germany from 02.08.2012 - Az. 27 F 96/11 [228].

[566] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[567] The American English spelling from original work is retained for terms and quotes.

9 — Analysis of existing technical confidentiality and cryptographic privacy definitions

Overview of Chapter 9

Chapter 9 discusses existing computer science definitions of the term 'confidentiality' with a focus of its usage and meaning in respect to the removal of data. To provide a sophisticated level of confidentiality is a main feature of an MSS:

> After an authorized modification the modified message and its still valid signature MUST not reveal the original data for those blocks which have been subject to the subsequent modification. Any authorized modification MUST hide the original value of the legitimately altered block, such that it cannot be recovered from the modified message and its valid signature.

This feature has been termed *privacy* in the MSS literature. Of course, this shall hold even if the adversary has access to the adjusted signature and the legitimately altered message. Even, if the adversary has access to many of those message and signature pairs or even if the adversary himself can instruct/use a legitimate entity to produce them based on input of the adversary's adaptive choice. Only if privacy is protected the applications of an MSS will allow to protect the confidentiality of trade secrets or personal data by a legitimate subsequent modification. The level of cryptographic confidentiality protection that is needed to fulfil the legally necessary protection to safeguard personal data or protect valuable trade secrets will be extracted. other than the mirror chapter on integrity (Chapter 5), this chapter starts with the cryptographic privacy definitions of selected existing SSS and RSS literature are analysed for their subtle differences and the resulting confidentiality protection.

- Section 9.1 shortly revisits definitions from the MSS literature to set the scene what the notions of privacy in the area of MSS encapsulates. Namely, this section will present the MSS goal by presenting the definitions stated by *Miyazaki*, *Johnson et al.*, *Ateniese et al.*, *Brzuska et al.*, and by *Agrawal et al.*.
- Section 9.2 analyses different *technical* definitions of the term confidentiality. The analysis includes standards like ISO 27000, ISO 7498-2, RFC 4949, US FIPS 199. Furthermore, an influential cryptographic security notion of confidentiality, known better under the term of indistinguishability secure against an adaptive chosen-ciphertext attack or IND-CCA2 is analysed in order to better understand cryptographic confidentiality. The standards were selected based on their adoption and widespread usage as a reference, and also due to their visibility towards governmental bodies[568].
- Section 9.3 describes what technical definitions of redaction offer towards a description of the needed strength of the MSS's goal of privacy / confidentiality. This includes the analysis of ISO 29100, ISO 27050, ISO 27042, ISO 27040, ISO 27038, and NIST SP 800-88.
- Section 9.4 presents the results of Chapter 9.

[568] See for example the statement from ISO 29100:2011 "Some jurisdictions might require compliance with one or more of the documents referenced in ISO/IEC JTC 1/SC 27 WG 5 Standing Document 2 (WG 5 SD2) — Official Privacy Documents References [...], but this International Standard is not intended to be a global model policy, nor a legislative framework." [250].

9.1. Definitions of privacy protection in selected existing MSS literature

This section will take a look at the selected influential literature on RSS and SSS that was already selected for the analysis of the integrity notion (in Chapter 5's Sec. 5.4). Namely, *Miyazaki et al.* [347, 349], *Steinfeld et al.* [455], *Johnson et al.* [273], *Agrawal et al.* [4] and *Ateniese et al.* [12]. Additionally, it will consider the first formalisation given by *Brzuska et al.* [64] for the SSS properties as defined in the work of *Ateniese et al.* [12]. It continues the introduction to privacy given in Sec. 3.9. From this work follows the extraction of the core confidentiality-like protection property of MSS used by this thesis: cryptographic privacy as defined by *Brzuska et al.* [64]. However, this also means that this section will state, or better *pre*-state, some of the harmonised definitions given in Chapter 11.

9.1.1. Privacy for RSS by *Miyazaki et al.*

In their work on RSS *Miyazaki et al.* describe the goal of privacy for an RSS as follows:

> "Privacy
> It is difficult for verifier to retrieve from a sanitized [redacted document] any information about sanitized subdocuments of the document." [349]

Note, the term 'sanitize' in the works of *Miyazaki et al.* [347, 349] refers to the action of redaction.[569] The authors further describe that the properties can be formally defined, and namely they specify that "Privacy [...] [is] defined by indistinguishability manner [...]"[570] [349]. The authors of [349] cite the work of *Goldwasser and Micali* [210] in relation to the indistinguishability. *Goldwasser and Micali* have built a scheme for which they "proved that extracting any information about the cleartext from the cyphertext is hard on the average for an adversary with polynomially bounded computational resources."[571] [210].

In an earlier paper from 2005 *Miyazaki et al.* [347] already formally describe the adversarial game, and thus the strength of the mechanism, for privacy. In a later paper from 2008 [349] *Miyazaki, Hanaoka, and Imai* detail this a bit more, their definition for privacy is as follows:

Definition 127 : Privacy by *Miyazaki et al.*

> *"Let DSDSS be a digitally signed document sanitizing scheme and $\mathcal{A}$ be a probabilistic polynomial-time Turing machine. $\mathcal{A}$ chooses a pair of documents M_0 and M_1 and position i such that $M_0[j] = M_1[j] (j \neq i)$ and $M_0[i] \neq M_1[i]$. Then $\mathcal{A}$ receives a signed document for M_b, where b is chosen randomly, in which the subdocument in position i has been [redacted]. DSDSS is* ***indistinguishable****, if for any $\mathcal{A}$ which has running time at most t the probability of the success of $\mathcal{A}$'s guess for b is $\frac{1}{2} + \epsilon$ here ϵ is negligible."*[572] [573] *[349]*

Miyazaki et al. **formally describe the level of protection of privacy pointing at the cryptographic notion of indistinguishability.** *Miyazaki et al.* in [349] cite the paper by *Goldwasser and Micali* [210] in which the notion of indistinguishability itself is described, but not yet named by that term in this paper from 1984. As it is fundamental to MSS privacy, the cryptographic indistinguishability notion is introduced in Sec. 9.2.4.

Miyazaki et al., based on the notion of the indistinguishability, state the level of strength against an probabilistic polynomial-time (PPT) attacker. This adversarial game got described in high-level language in Sec. 3.9. This thesis offers the slightly adapted formalisation in harmonised notation in Sec. 11.13.3.

[569] See "Sanitized (i.e., masked)" [347, p. 242] and "The administrative office then deletes (masks) sensitive data such as personal information or national secrets and discloses a sanitized version of the document." [349] for *Miyazaki et al.*'s interpretation of the functionality of 'sanitization'.

[570] The obvious spelling mistake of "[...] indistinguishablility [sic] manner [...]" [349] was corrected.

[571] The American English spelling from original work is retained for terms and quotes.

[572] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[573] For clarity the word "sanitized" has been replaced by the notion of redaction used in this thesis. See the discussion at the beginning of this section and also Footnote 569.

9.1.2. Privacy for RSS by *Steinfeld et al.*

In the work from *Steinfeld, Bull, and Zheng* [455] (extended version in [457]) the authors identify the need for an RSS to "[...] exclude the possibility that the extracted signature may in general leak some information on the content of the unextracted (deleted) submessages." [455]. Before the above description of the goal of the privacy notion, they state that "[u]nlike standard signatures, where unforgeability is the only security requirement, many applications [...] may also have a **privacy** security requirement."[574] [455]. They offer the following informal definition:

Definition 128 : Privacy by *Steinfeld et al.*

> *"It is infeasible for an attacker to obtain, given an extracted signature σ for a subdocument M' of some document M, any information on any submessage in the original document M which has not been extracted [(i.e., copied over from the original)] into the subdocument M' [...]" [457].*

Their formal notion is defined alike an indistinguishability notion, in line with other privacy notions in the MSS literature.

The adversary from *Steinfeld et al.* knows the Signer's secret key. Note that the adversary described by *Steinfeld et al.* [457] is strong: Not only can it adaptively choose both input messages m^1 and m^2 as well as a corresponding ADM, but it even has "[...] knowledge of the signer's secret key." [457]. The key generation, as well as generation of signatures and redactions are done by the oracle; they all run genuinely.

***Steinfeld et al.* formally describe the level of protection of privacy using a game-based definition based on the cryptographic notion of indistinguishability.** Re-explain using the harmonised notation their definition is as follows: The adversary against privacy can define two messages (m^1 and m^2) which differ only in some block. W.l.o.g. one block at the same position is assumed to differ and denoted as i^*, i.e. $m^1[i^*] \neq m^2[i^*]$. The adversaries ADM is the same for both messages and must capture all differentiating parts, i.e. $\mathsf{ADM}_m(i^*) = 1$. A left-or-right redact oracle is now either redacting the first or the second of the inputs at all differentiating block, here $m[i]$, and the adversary has to guess which message was redacted. The adversary wins and breaks privacy if it can guess correctly with a non negligible advantage over $\frac{1}{2}$. See the original publications for more details on their game-based definition [455, 457]. *Steinfeld et al.*'s adversarial game was taken into account when defining the standard privacy notion of this thesis. See Sec. 3.9 and However, the notion of standard privacy in Sec. 11.13.3 is adapted. Namely, it is weaker than *Steinfeld et al.*'s as it **dis**allows the adversary to know $\mathsf{sk}_{\mathrm{sig}}$.

9.1.3. Privacy for RSS by *Johnson et al.*

Johnson, Molnar, Song, and Wagner describe the goal of the "important"[575] [273] privacy property for their RSS as follows: "[...] redacted signatures do not reveal the redacted parts of the original message. We can see that they reveal the locations that have been redacted, much like typical redaction of paper documents does, but this is all that is leaked." [273] The authors further identify — what they call — "a serious limitation: [...] a scheme will conceal the presence of deletions, which introduces the risk of semantic attacks." [273]. The example of such an attack from the original paper has been depicted in Fig. 22 in the high-level introduction to privacy given in Sec. 3.9. The above codifies not only the need to offer 1CD[576] detection to identify all occurred subsequent authorized modifications. It further codifies that the scheme needs to support a detection level that allows redaction detection on each of all individual blocks, which equals to BCD as defined in Sec. 6.3.1[577]. If the structural integrity protection protects the position, e.g., for a linear document, and a private scheme offers BCD this is equal to the protection sought after by *Johnson et al.*: "[A] signature $\mathsf{Sig}(w)$ on a redaction w of x may legitimately reveal which portions of x were deleted, and specifically, the presence and location of redacted segments is protected from tampering by the signature. However, it does not reveal the previous contents of the redacted portions of the document." [273].

[574] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

[575] The authors stress, as well as this thesis, "that this privacy property may be important to many applications." [273]

[576] 1CD means at least one subsequent modification is detected, as defined in Sec. 6.3.1.

[577] BCD means all individual block occurred change detection. As a redaction is the only change possible, knowing that a block has been modified (as in BCD) in an RSS means it has been redacted.

Regarding the strength of the privacy property they note in reference to *Bellare, Goldreich, and Goldwasser* work that their own "[...] notion of privacy for redactable signatures has a parallel in their notion of privacy for incremental signatures [40]." [273] Turning to *Bellare et al.*'s paper [40], privacy is concerned with hiding "[...] the information regarding previous versions of the document which can be inferred by the legitimate party when inspecting the current document together with the current cryptographic form." [40]. As an explanatory example, they state that privacy hides the contents of D' not in D when " [...] given a document D together with its (updated) cryptographic form and we are told that D was obtained from some other document, called D', by a deleting a single symbol." [40]. Further *Bellare et al.* describe that privacy can come in two different strengths: perfect and partial.

Definition 129 : Perfect and partial privacy by *Bellare et al.*

> *"[...] [G]iven a document D together with its (updated) cryptographic form and [...] told that D was obtained from some other document, called D′, by a deleting a single symbol." [40]*

> *"Perfect privacy would mean that we cannot tell the location of the deleted symbol. [...] Specifically, given a document D and a signature to it, it should be infeasible to distinguish whether the signature was by the document system in response to a create command or in response to a text modification command." [40]*

> *"Partial privacy may mean that we cannot tell the identity of the deleted character (but we may have some information regarding its location)." [40]*

Bellare et al.'s definition of partial privacy means that one can detect subsequent modifications, which is 1CD, and eventually further their location equal or potentially better than BCD, as in the definition by *Johnson et al.*. Regarding formal definitions to judge the cryptographic strength of privacy *Johnson et al.* make no statements in [273] and *Bellare et al.* offer in [40] the following: "A definition of perfect privacy can be easily produced following the standard paradigms. Specifically, given a document D and a signature to it, it should be **infeasible to distinguish** whether the signature was by the document system in response to a create command or in response to a text modification command. Definitions of partial privacy may vary for ones in which the amount of modification is the only information being leaked to ones in which only the secrecy of replaced/deleted symbols is preserved."[578] [40]. For the notion termed 'perfect privacy' the strength can be cryptographically obtained from this, and following the description it would include the notion of transparency as defined in the later work of [12] that was formalised by *Brzuska et al.*. For the latter notion of less strict 'partial privacy' the authors do not offer an explicit definition in [40].

***Johnson et al.* define the strength of privacy using *Bellare et al.*'s notion of "infeasable to distinguish" [40].** The privacy notion stated by *Johnson, Molnar, Song, and Wagner* in [273] means the adversary cannot efficiently distinguish between a subsequent modification of a signed document that was authorized and a modification that took place before the document got signed. This understanding is also in line with the privacy notion formally defined by *Steinfeld, Bull, and Zheng* [457]. Further, this understanding of privacy has been used as the basis for this thesis when giving the high-level presentation of the goal of privacy that depicts a black box in which the undistinguishable task is happening in Fig. 21 on 60.

Moreover, by stating that "[...] a redaction w of x may legitimately reveal which portions of x were deleted [...]" [273] the authors require that at least any subsequent modification is detected, i.e., 1CD detection as defined in Sec. 6.3.1. This thesis interprets this as hinting that a non-transparent and thus non-interactive public accountability (PUB) is helpful. The authors of [273] point to *Bellare et al.*'s notion of "[...] unfeasibility to distinguish whether the signature was by the document system in response to a create command or in response to a text modification command." [40]. However, the cryptographic strength of the notion of "partial privacy" [40], which better corresponds to *Johnson et al.*'s goal of detecting at least any subsequent modification, i.e., offer ≥1CD integrity protection, is not formally defined in neither [40] nor [273].

[578] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

Note, the notion of strong privacy defined by *Bellare et al.* is not used in this thesis. In this thesis *perfect privacy* means private in an information-theoretical sense. This perfect privacy notion is defined in this thesis and presented in Proof 33 when presenting $struct\mathcal{RSS}$ in Sec. 14.10 and also in Proof 40 when presenting $merge\mathcal{RSS}$ in Sec. 14.12.

9.1.4. Privacy for SSS by *Ateniese et al.*

In the original work that introduced the notion of sanitizable signatures as used in this thesis, *Ateniese, Chou, de Medeiros, and Tsudik* state that the "blacked-out sections" [12] shall "remain confidential" [12]. They describe the confidentiality requirement of the privacy property further as follows:

Definition 130 : Privacy by *Ateniese et al.*

> *"Given a sanitized signed message with a valid signature, it is **impossible** for anyone (except the signer and the censor) **to derive any information** about the portions of the message that were sanitized by the censor. In other words, all **sanitized information is unrecoverable**.*$^{+}$
> $^{+}$ *Unless of course the original message is stored by the signer and/or the censor."*[579] *[12]*

Further, they differentiate the confidentiality protection termed 'privacy' from the detectability of authorized occurred or future modifications termed 'transparency'. The term transparency that also got carried over by later SSS and RSS literature, unfortunately has some duality in the meaning. In natural and legal language usage it means the exact opposite. This observation led to Analysis Result 23: Cryptographic and legal notion of transparency have an opposite meaning (see Sec. 9.4.4).

Regarding the need for transparency the authors note "strong transparency is not always better. In certain circumstances, weak transparency is actually preferable. For example, if a document originally signed by some government official is later released by a certain government agency — acting as a censor — under the Freedom of Information Act, the general public would likely prefer knowing which parts of the document could have been sanitized." This can be interpreted as *Ateniese et al.* acknowledging the need for a non-interactive public form of detectability and thus accountability (PUB) of subsequent authorized modifications. However, due to a problem in their transparency property stated in [12], their concrete definition of the notion is not further discussed. This thesis result got also jointly published together with *Samelin and Posegga* [391] (see Appendix A publication n^{o} 6). The details of this problem are presented in Sec. 11.6.4.

***Ateniese et al.*'s definition of privacy for the redaction describes that it is impossible to derive any information about redacted block.** The definition for privacy was the one that got formalised later by *Brzuska et al.* [64] for SSS. The formal version is thus analysed next in Sec. 9.1.5. *Ateniese et al.* make very clear that "sanitized information is unrecoverable" [12] and thus raise the bar for the privacy protection, without giving a more formal description of it.

They differentiate between the property of privacy and the property of detectability of authorized occurred or future modifications. The latter they termed 'transparency'. Their "strong transparency [...] guarantees that the verifier does not know which parts of the message are immutable and thus does not know which parts of a signed message could potentially be sanitizable." [12]. This maps to what this thesis described as future change detection (*FD levels) in Sec. 6.3.2. A strongly private SSS following *Ateniese et al.* would require to make it invisible for the Verifier to detect which blocks are admissible and which are not. This maps to the detectability aspect of integrity denoted as NFD[a]. Their definition of 'transparency' and its problems are discussed in more detail in Sec. 11.6.4.

Interestingly, *Ateniese et al.*'s notion for an invisible sanitizable signature scheme, i.e., a scheme that offers NFD, got neither formalised in [12] nor picked up by [64]. As the property remained of interest it was recently formalised and a provable construction was published at PKC 2017 as joint work together with *Camenisch, Derler, Krenn, Samelin, and Slamanig* [107] (see Appendix A publication n^{o} 55).

[a] NFD stands for *no future change detection* in this thesis as defined in Sec. 6.3.2.

[579] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

9.1.5. Privacy for RSS and SSS by *Brzuska et al.*

The works of *Brzuska et al.* have formalised a number of properties, among them is the privacy property for SSS and RSS. For the realm of SSS the work of *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk* from 2009 [64] was the first to formalise the property for SSS as described in the work of *Ateniese et al.* [12]. In 2010 *Brzuska, Busch, Dagdelen, Fischlin, Franz, Katzenbeisser, Manulis, Onete, Peter, Poettering, and Schröder* published their work on RSS for tree structured data [66]; therein they formally described a privacy property of the same strength for the RSS domain.

9.1.5.1. Privacy for SSS by *Brzuska et al.*

For SSS *Brzuska et al.* stated the privacy goal for subsequently modified messages as follows:

> "PRIVACY. Sanitized messages and their signatures should not reveal the original data (i.e., the parts which have been sanitized)." [64]

They as well based their notion "on the indistinguishability notion for encryption" [64], like *Johnson et al.* [273], *Miyazaki et al.* [347, 349], or *Agrawal et al.* [4]. The notation used for the harmonised presentation of security properties for MSS is inspired by the works of *Brzuska et al.*. In the harmonised notation and with some additions and fixes from the existing body of work *Brzuska et al.*'s definition for privacy of an SSS is as follows (see Sec. 11.6.3 for further details):

Definition 156 : Standard Privacy for SSS following *Brzuska et al.*

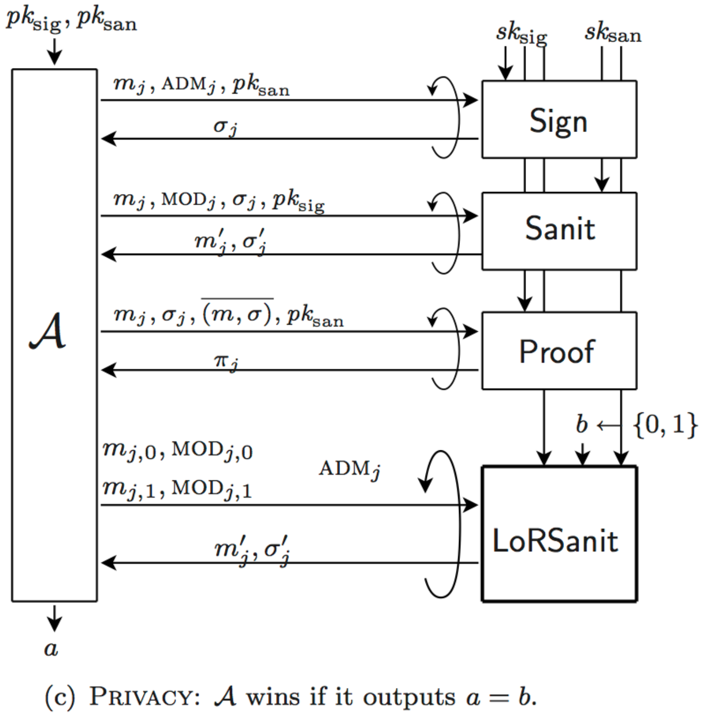
*An SSS is **private**, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 51 returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ).*

```
Experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{pk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
  $b \overset{\$}{\leftarrow} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}})}_{\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{LoRSanit}(1^\lambda,\ldots,\mathsf{sk}_{\mathrm{sig}},\mathsf{sk}_{\mathrm{san}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
    where oracle LoRSanit on input of $m_0, \mathsf{MOD}_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$ // resulting $m'$ is the same
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{ADM} \not\subseteq m_0$ or $\mathsf{ADM} \not\subseteq m_1$ or $\mathsf{MOD}_0(\mathsf{ADM}) \not\subseteq \mathsf{MOD}_0(m_0)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}) \neq \mathsf{MOD}_1(\mathsf{ADM})$, return $\perp$
      let $(m_i, \sigma_i) \leftarrow \mathsf{Sign}(1^\lambda, m_b, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{ADM})$
      return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_i, \mathsf{MOD}_b, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$
  return 1, if $a = b$
```

Figure 51. Standard Privacy Experiment for SSS based on *Brzuska et al.* [64] including additions from *Krenn et al.* [294]
(same Figure as on page 265)

Further to that the authors note that in their notion of privacy "[...] the adversary does not get to choose the signature σ_b for inputs to the left-or-right box [denoted as oracle LoRSanit in Fig. 51]. Instead, this signature is first computed from scratch."[580] [64]. The adversary, which is polynomially bounded in its time to attack, can repeatedly ask the LoRSanit oracle for new challenges. This is depicted nicely in the original work as reproduced in Fig. 42. Fig. 42 also underlines, that during the whole runtime the adversary has access to a signing oracle that allows to generate genuine signatures; a sanit oracle that sanitizes for the adversary and a proof oracle that generates valid proofs that the adversary can use to check the accountability in an interactive and non-public mode (INT as described in Sec. 6.4.1 on page 150).

[580] The American English spelling from original work is retained for terms and quotes.



$pk_{\text{sig}}, pk_{\text{san}}$
sk_{sig}
sk_{san}
m_j, ADM$_j$, pk_{san}
σ_j
Sign
m_j, MOD$_j$, σ_j, pk_{sig}
m'_j, σ'_j
Sanit
$\mathcal{A}$
$m_j, \sigma_j, \overline{(m,\sigma)}, pk_{\text{san}}$
π_j
Proof
$b \leftarrow \{0,1\}$
$m_{j,0}$, MOD$_{j,0}$
$m_{j,1}$, MOD$_{j,1}$
ADM$_j$
m'_j, σ'_j
LoRSanit
a
(c) PRIVACY: $\mathcal{A}$ wins if it outputs $a = b$.

Figure 42. Figure from *Brzuska et al.* [66] showing that the adversary can adaptively and repeatedly query all oracles

The authors note that the adversary does not get to see the internal σ_i nor the non sanitized m_i. Exactly, this behaviour of the LoRRedact oracle was the basis for the high-level description of the privacy property as visualised in Fig. 21 in Sec. 3.9. The given attacker model makes especially sense as in many scenarios the attacker can query automated systems that emit signed documents or do custom sanitizations and thus it is very safe to assume that the adversary has access to many signed documents and sanitized versions. Note especially that generating signatures protects a document's authenticity, i.e. integrity and authentic origin, and not per-se its confidentiality. It is the additional step of sanitization inherent to malleable signature schemes that shall protect the original's or previous version's contents. Schemes not meeting the privacy property by *Brzuska et al.* [66] can be attacked and thus can be a limitation as mentioned in Sec. 12.1 as Shortcoming 1.

9.1.5.2. Privacy for RSS by *Brzuska et al.*

For RSS *Brzuska, Busch, Dagdelen, Fischlin, Franz, Katzenbeisser, Manulis, Onete, Peter, Poettering, and Schröder* stated in [66] the privacy goal for subsequently modified messages as follows:

> "PRIVACY. No one should be able to gain any knowledge about parts cut off the tree from its structural signature without having access to these parts. Our definition is similar to the standard indistinguishability notion for encryption schemes." [66].

Their experiment to model the adversary capabilities and its goal is as follows in the original paper for trees:

In Fig. 43 L denotes the subtree that shall be redacted from the tree T; then $T \setminus L$ denotes the remaining tree after a successful redaction; the algorithm for redaction was called sCut in the original work. As for SSS their game based description of the property is based on indistinguishability of the two inputs after the redaction, which shows in the game. "This is formalized by demanding that, given a subtree with a signature and two possible source trees, one cannot decide from which source tree the subtree stems from. Intuitively, it follows that one cannot derive any information about the cut parts."[581] [66]. Like in the SSS's privacy game (see Fig. 51 on page 265), the probability for an efficient adversary that the experiment LeakPriv will evaluate to 1 shall be negligibly close to $\frac{1}{2}$, i.e., the adversary shall not be better than guessing.

[581] The American English spelling from original work is retained for terms and quotes.

```
Experiment $\mathsf{LeakPriv}_{\mathcal{A}}^{strucSig}(\lambda)$
    $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
    $b \overset{\$}{\leftarrow} \{0,1\}$
    $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk_{sig}},\cdot),\mathsf{SignCut}(\cdots,\mathsf{sk_{sig}},b)}(1^\lambda, \mathsf{pk})$
        where oracle SignCut
        for input $T_0, L_0, T_1, L_1$:
            if $T_0 \setminus L_0 \neq T_1 \setminus L_1$ abort
            let $(T_b, \sigma_b) \leftarrow \mathsf{Sign}(1^\lambda, T_b, \mathsf{sk_{sig}})$
            return $(T_b', \sigma_b') \leftarrow \mathsf{Redact}(1^\lambda, T_b, L_b, \sigma_b, \mathsf{pk_{sig}})$.
    return 1, if $d = b$
```

Figure 43. Privacy Experiment for RSS on tree structured data from *Brzuska et al.* [66]; the algorithm names have been slightly changed and an index j was removed to simplify the presentation compared to [66]

For this thesis, the above definition got interpreted for non tree structured data. Note, the original work of *Brzuska et al.* did assume that all elements in the set T are redactable and thus does not contain any notion of admissible redactions.[582] If one adds this into the harmonised notation a privacy for RSS *Brzuska et al.*'s definition then can be stated as follows (see Sec. 11.13.3 for further details):

Definition 174 : Standard Privacy for RSS

*An RSS is **private**, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$ given in Fig. 60 returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ).*

```
Experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$
    $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
    $b \overset{\$}{\leftarrow} \{0,1\}$
    $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk_{sig}},\cdot),\mathsf{LoRRedact}(1^\lambda,\cdots,\mathsf{sk_{sig}},b)}(1^\lambda, \mathsf{pk})$
        where oracle LoRRedact on input $m_0, \mathsf{MOD}_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}_0, \mathsf{ADM}_1$:
            if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$
            if $\mathsf{MOD}_0 \nsubseteq \mathsf{ADM}_0$, return $\perp$
            if $\mathsf{MOD}_1 \nsubseteq \mathsf{ADM}_1$, return $\perp$
            if $\mathsf{ADM}_0 \nsubseteq m_0$ or $\mathsf{ADM}_1 \nsubseteq m_1$
                or $\mathsf{MOD}_0(\mathsf{ADM}_0) \nsubseteq \mathsf{MOD}_0(m_0)$ or $\mathsf{MOD}_1(\mathsf{ADM}_1) \nsubseteq \mathsf{MOD}_1(m_1)$, return $\perp$
            if $\mathsf{MOD}_0(\mathsf{ADM}_0) \neq \mathsf{MOD}_1(\mathsf{ADM}_1)$, return $\perp$
            let $(m_i, \sigma_i) \leftarrow \mathsf{Sign}(1^\lambda, m_b, \mathsf{sk_{sig}}, \mathsf{ADM}_b)$
            return $(m', \sigma') \leftarrow \mathsf{Redact}(1^\lambda, m_i, \mathsf{MOD}_b, \sigma_i, \mathsf{pk_{sig}})$.
    return 1, if $a = b$
```

Figure 60. Standard Privacy Experiment for RSS based on *Brzuska et al.* [66] including additions from *Krenn et al.* [294]
(same Figure as on page 289)

9.1.5.3. Analysis *Brzuska et al.*'s privacy definitions

Additionally to formal definitions of privacy, the works of *Brzuska et al.'s* define also the stronger privacy notion of transparency formally for RSS and SSS as well. However, they also state that "[...] for various application examples privacy is in fact sufficient, namely in all cases, where the receiver already expects partly sanitized documents." [66] Thus, the authors in [66] for RSS and in [64] for SSS declare that privacy is the important property, and that it "by using a private, non-transparent scheme, one thereby gains [...] efficiency." [66]

[582] This is in line with other works of that time, e.g. *Johnson et al.* [273].

***Brzuska et al.*'s definition of privacy for SSS and RSS are formally defined and capture a good minimum confidentiality protection.** In their works [64, 66] formal definitions are presented based on adversary models. The level of confidentiality protection chosen is a good balance and includes the goals captured in existing works, e.g., "partial privacy" [40] as mentioned by *Johnson et al.* [273] and it is also based on indistinguishability like the notion from *Miyazaki et al.* in [349] and *Steinfeld et al.* in [455, 457]. With respect of the knowledge of keys *Brzuska et al.* does not allow the adversary access to the any, especially not the Signer's secret key. Here, the definition of *Steinfeld et al.* was stronger. As it was build upon it, it incorporates the privacy notion of the work of *Ateniese et al.* [12].

The authors differentiate a stronger privacy property, termed transparency, from that of privacy. However, they do not see transparency as an essential and required property for a secure scheme. A secure scheme for them is private, unforgeable and accountable. A secure scheme that additionally fulfils *Brzuska et al.'s* definition of transparency can thus be mapped to ACA – UCD – INT[a] protection, as defined in Chapter 6. Their definition of transparency for SSS is discussed in more detail in Sec. 11.6.4. Their "[...] strong transparency [...] guarantees that the verifier does not know which parts of the message are immutable and thus does not know which parts of a signed message could potentially be sanitizable." [12]. This can be mapped to what this thesis described as future change detection (*FD levels) in Sec. 6.3.2. A strongly private SSS following *Ateniese et al.* would require to make it infeasible for the Verifier to detect which blocks are admissible and which are not, which would be NFD[b] following the definition given in Sec. 6.3.2. Their definition of the notion of 'transparency' is discussed in more detail in Sec. 11.6.4.

Due to *Brzuska et al.'s* privacy and transparency notion being formally defined and because it captures existing notions from the existing body of work, the thesis selected it as a starting point, e.g., see Sec. 11.6.3 and Sec. 11.13.3. *Brzuska et al.* show that their notion is strong and based on an indistinguishability notion comparable to the indistinguishability for encryption schemes (see Sec. 9.2.4 for a definition). The same approach was taken by other works, e.g., *Steinfeld et al.* [457]. Thus this notion provides a well defined strength of confidentiality and a starting ground for a definition of privacy. A stronger notion, as well based on indistinguishability, that allows access to the Signer's secret signature generation key, was formalised in *Steinfeld et al.* for RSS. While this is not the case in the SSS definitions of privacy. The weaker notion from *Brzuska et al.* is used in this thesis instead, as their work showed, that it is enough to offer a good resilience against attacks.

Finally note that *Brzuska et al.'s* notation is used for the harmonisation of existing MSS and new properties, e.g., the notation of the thesis for games presented in Sec. 3.8.1 follows the notation found in their works, e.g., [64, 66].

[a] Meaning authorized changes are allowed (ACA) by the integrity protection mechanisms and it offers the detection if any unauthorized change occurred (UCD) with non-public interactive accountability (INT).

[b] The mechanisms offer no detection of the potential of future subsequent changes (NFD).

9.1.6. Privacy for SSS by *Agrawal et al.*

In their 2009 work *Agrawal, Kumar, Shareef, and Rangan* [4] state the following: "A sanitizable signatures [sic] protects the **confidentiality** of a specified part of the document while ensuring the integrity of the document."[583] [4]. Instead of a security property named privacy, they define in their paper directly an indistinguishability property. Basing privacy on indistinguishability is in line with the other MSS works' privacy notions. *Agrawal et al.* defines it as follows:

> "We have the following game Exp_{ind} for indistinguishability:
>
> 1. The simulator S gives $param$ and PK to the adversary A.
> 2. A is allowed to query the signing oracle q_s times adaptively. The oracle is the same as the one in the game for unforgeability.
> 3. A sends two different signatures σ_0, σ_1 on M_0, M_1 respectively and a sanitized message M', where M' differs from M_0 and M_1 only at bits that are allowed to be sanitized.

[583] **Bold** face and other *emphasis* like ~~deletions~~ or <u>underlining</u> have been added for highlighting.

> 4. S picks a random bit b and sends σ'_b to A which is the signature obtained from the sanitization of message M_b.
> 5. Finally, A outputs bit b'.
>
> A wins the game if $b = b'$. The advantage of A is $|Pr[b = b'] - \frac{1}{2}|$.
>
> **Definition.** A sanitizable signature scheme is said to be unconditionally indistinguishable if there is no adversary winning the above game with advantage greater than 0 with any number of queries to the signing oracle."[584] [4]

This is close to, but not the same as the one by *Brzuska et al.* — termed *standard privacy* in this thesis (see the subsection above and Sec. 11.6.3 for further details). The main difference towards standard privacy is that *Agrawal, Kumar, Shareef, and Rangan*'s adversary (in Step 3) also supplies, hence controls, the signatures on the two input messages (σ_0, σ_1) to the Left-or-Right-Sanitize oracle. Of course the adversary cannot generate the two signatures itself, as it does not know $\mathsf{sk}_{\mathsf{sig}}$, it uses a signing oracle for this. This is in contrast to the standard privacy by *Brzuska et al.* given in [64]; the authors of [64] noted in that regards explicitly: "[...] that the adversary does not get to choose the signature σ_b for inputs to the left-or-right box. Instead, this signature is first computed from scratch. This corresponds to the "hospital setting" mentioned in the introduction, where the medical data and, in particular, their signatures are kept confidential and only the sanitized document is released. One may define a stronger version where the adversary gets to choose σ_0, σ_1, but it seems much harder to realize this requirement efficiently."[585] [64]. Hence, the privacy property formalised by *Agrawal et al.* [4] is stronger than standard privacy from Definition 156.

Regarding stronger privacy properties *Agrawal et al.* also describe the property of transparency with "various levels" [4]. Their definitions of transparency are not individually formalised in [4], which leads to the following small deficiencies in their notion: The authors give the following definition for a negated transparency property: "No transparency. The verifier knows which part of the document is sanitized [...]" [4]. Negating a property that is formally defined using probabilistic notions, like the transparency property, obviously means that the scheme offers no form of transparency. However, this negation will not be enough to achieve the functionality of detecting occurred changes in all individual blocks (BCD[586]) that *Agrawal et al.* adequately describe by stating that "[t]he verifier knows which part of the document is sanitized [...]" [4]. To realise such a functionality requires a mechanism that offers integrity protection with block-level accountability and public and non-interactive accountability (denoted BCD – PUB), as described in Sec. 6.5 and in Sec. 6.4.3. Only then the Verifier knows with overwhelming probability which block or blocks of the document are sanitized. This is formally described and proven later in Sec. 13.1.3.

Agrawal et al.'s notion of "strong transparency" [4] is not formally defined in [4], but judging from its informal description it is assumed to follow the same goal as the one formalised by *Brzuska et al.*, which can be found in Definition 158 (see Sec. 11.6.4): "The verifier does not know if the message has been sanitized [...]" [4] is comparable to *Brzuska et al.*'s statement that "[i]t should be infeasible to decide whether a message has been sanitized or not [...]" [64].

Agrawal et al.* define a privacy notion stronger than standard privacy defined by *Brzuska et al.
The privacy property formalised by *Agrawal et al.* is stronger than standard privacy defined by *Brzuska et al.*, which can be found in Definition 156. It is based on the indistinguishability of two input messages and their valid signatures that can both be made into the same output message by sanitization.
Due to the varying strength, this thesis will analyse if the privacy definition of *Brzuska et al.* is strong enough for the confidentiality protection or if a stronger one, like the one from *Agrawal et al.*, is legally required. The thesis offers an analysis how existing schemes compare in terms of privacy and what attacks are possible if the strength of *Brzuska et al.* is not given. The latter analysis results are offered in Shortcoming 1 in Sec. 12.1. The legally required strength is given in Sec. 10.1 as Requirement 9.

[584] For increased readability the original numbering of the definition was removed. Further, an additional closing square bracket in the calculation was considered a typo and got removed.

[585] The American English spelling from original work is retained for terms and quotes.

[586] This was defined in Sec. 6.3.1 on page 145.

The authors of [4] further differentiate between the property of privacy and the property of cryptographic transparency, but they do not provide formal definitions of both individually. Instead their notions are both integrated into their indistinguishably notion. As such, *Agrawal et al.* [4] do not see transparency as optional, but as a required part of privacy. Thus, this thesis sees both of their private and transparent SSS constructions as offering ACA – UCD – INT integrity[a].

[a] Meaning authorized changes are allowed (ACA) by the integrity protection mechanisms and it offers no detection of all subsequent changes but only of unauthorized ones (UCD) with non-public interactive accountability (INT). UCD-level detectability is due to strong transparency, which in turn means that the SSS can only achieve INT-level accountability and not PUB-level.

9.2. Confidentiality definitions with respect to MSS's privacy protection against recoverability

This section analyses a set of confidentiality definitions from technical standards: ISO 2700x, ISO 7498-2, RFC 4949, and US FIPS 199. It concludes with a presentation of the cryptographic notion of IND-CCA2. The goal is to identify the technical view on the notions of confidentiality, especially to find a sound description of its cryptographic strength against attacks. Whenever possible the analysis tries to find a relation between the cryptographic strength or the application of confidentiality protection and data protection regulation, e.g., confidentiality by encryption or removal to safeguard the privacy of personal data. Note, that in the latter the 'privacy' term's meaning relates to the legal terminology of data protection (see for example Sec. 8.2.1) rather than the cryptographic feature termed *privacy* in the MSS literature (see previous Sec. 9.1).

9.2.1. Definition of confidentiality from ISO 2700x and ISO 7498-2

In the ISO standard 27000 the term 'confidentiality' is described as a property that applies to information as follows:

Definition 131 : Confidentiality from ISO 27000

> *"**confidentiality***
> *property that information is not made available or disclosed to unauthorized individuals, entities, or **processes** [...]" [249]*

About the same wording[587] can also be found in ISO 7498-2 from the year 1988 [239].

ISO 27000 is about the vocabulary of the ISO 2700x family[588]. On the requirements of cryptographic controls ISO 27001 mentions as the objective "[t]o ensure proper and effective use of cryptography to protect the confidentiality, authenticity and/or integrity of information" [252, p. 14]. However, ISO 27001 let alone the whole 2700x series is not meant to offer further technical details. Towards this ISO 27001 states that "[...] controls shall be selected and implemented to meet the requirements identified by the risk assessment and risk treatment process. This selection shall take account of the criteria for accepting risks [...] as well as legal, regulatory and contractual requirements" [246, p. 5].

At the most specific level of the ISO 2700x series ISO 27000 lists so called sector-specific guidelines standards [260, Fig. 1, p. 21] as in the scope of the series. One of them that is obviously related to data protection is ISO 27018 with the title stating "Code of practice for protection of personally identifiable information (PII) in public clouds acting as PII processors" [255]. In ISO 27018 the term 'confidentiality' gets mentioned in relation to non-disclosure agreements which does not relate to the use of cryptography[589]. In more detail ISO 27018 prescribes encryption, e.g., as a control for communication over "public data-transmission networks" [255, p. 20] of PII, however it does also not elaborate on its strength. It does mention deletion in regards to the data subject's rights, however it does not mention redactions or other techniques that would relate to the way MSS protect the confidentiality through removal.

[587] "The property that information is not made available or disclosed to unauthorized individuals, entities, or processes." [239].

[588] "Provides background, terms and definitions applicable to the ISMS Family of Standards" [249]

[589] "Individuals under the public cloud PII processor's control with access to PII should be subject to a confidentiality obligation. " [255].

Confidentiality in ISO 2700x is the property that information disclosure to unauthorized parties is prohibited. The ISO series 2700x defines the notion as well as their goals on a very high-level. It does not offer any indication of its technical strength. A link to personal data protection is made as the analysed ISO 27000 contains a pointer to ISO 27018. The linked ISO 27018 mentions "data minimization"[a] [255] as well as "confidentiality and non-disclosure agreements" [255] or "deletion" [255]; however it provides no indication on the strength of the confidentiality protection.

[a] The American English spelling from original work is retained for terms and quotes.

9.2.2. Definition of confidentiality from RFC 4949

Additionally to referencing above stated definition from ISO 7498 (see Definition 131) the RFC 4949 offers the following definition:

Definition 132 : Data Confidentiality in RFC 4949

> *"1. (I) The property that data is not disclosed to system entities unless they have been authorized to know the data. ([...] Compare: privacy.)" [438]*

Note, the above definition gives a hint that the notion of data confidentiality (or just confidentiality[590]) should not be confused with the notion of privacy. Namely, RFC 4949 states under the privacy definition that one "SHOULD NOT use this term [privacy] as a synonym for "data confidentiality" [...]" [438].

This thesis uses 'privacy' to describe a data confidentiality related security property to stay in line with existing MSS literatures' terminology. This thesis generally agrees that "[p]rivacy is a reason for security rather than a kind of security." [438]. Still, to be consistent with existing literature this thesis decided to stay with the term of cryptographic privacy to describe the confidentiality goal inherent to the concept of MSS. Especially after the formalisation of the property by *Brzuska et al.'s* in [64, 66] the term has been generally agreed to be used to describe the confidentiality offering of MSS, e.g., by *Slamanig and Rass* [441], *Canard et al.* [112], *Attrapadung et al.* [15] , or *Kundu and Bertino* [299].

RFC 4949 describes data confidentiality as the property that information disclosure to unauthorized parties is prohibited. RFC 4949 defines its notion on a high-level. Confidentiality shall protect against the disclosure of data "to system entities unless they have been authorized to know the data" [438].

9.2.3. US law: Confidentiality in FIPS 199 and NIST SP 800-122

Because US FISMA was also found helpful in the integrity analysis (see Sec. 4.3.4.2) it was analysed with respect to confidentiality. The US act FISMA defines three security objectives for information and information systems and points to FIPS 199 as the technical standard related to that legal text. FIPS 199 itself then defines confidentiality based on the United States Code — the general and permanent laws of the United States (U.S.C.) — as follows:

Definition 133 : Confidentiality found in FIPS 199 and NIST SP 800-122 quoting U.S.C.

> *"CONFIDENTIALITY*
> *"Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information [...] " [44 U.S.C., Sec. 3542]*
> *A loss of confidentiality is the unauthorized disclosure of information."[589] [354]*

The FIPS Publication 199 does not define any further technical mechanisms nor their strengths, but "levels of potential impact on organizations or individuals should there be a breach of security (i.e., a loss of confidentiality [...])"[589] [354]. Thus, it is not further analysed.

[590] RFC 4949 sees them as synonymous as it states: "confidentiality See: data confidentiality." [438]

Additionally, there is a special publication 800-122 by the US NIST on the "Confidentiality of PII" to offer guidance, including examples of what can constitute PII [332]. It states the same definition of confidentiality as FISMA, citing 44 U.S.C. § 3542. Noteworthy is that, alongside general information security safeguards, the guideline suggests to deploy "[p]rivacy-specific safeguards [...] for protecting the confidentiality of PII. These controls provide types of protections not usually needed for other types of data." [332]. Among them is the action of "[...] Suppressing the Data — Deleting an entire record or certain parts of records [...]" [332] which this thesis calls redaction. Other actions are the following: "[...] Generalizing the Data [...], Introducing Noise into the Data [...], Swapping the Data [...], Replacing Data with the Average Value [...]."[591] [332]. The goal of the actions including the deletion, but probably in combination, shall be that "[u]sing these techniques, the information is no longer PII, but it can retain its useful and realistic properties." [332]. Like the FIPS Publication 199 NIST's SP 800-122 makes no statements regarding explicit technical mechanisms nor their strengths. Instead it states that "[o]rganizations should apply the appropriate safeguards for PII based on the PII confidentiality impact level. Not all PII should be protected in the same way. Organizations should apply appropriate safeguards to protect the confidentiality of PII based on the PII confidentiality impact level. Some PII does not need to have its confidentiality protected, such as information that the organization has permission or authority to release publicly (e.g., an organization's public phone directory)."[591] [332].

FIPS 199 and NIST SP 800-122 quote from the general and permanent federal statutes of the United States (U.S.C.) which defines data confidentiality as the property that information disclosure to unauthorized parties is prohibited. Following the quote confidentiality is defined as "[...] preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information."[44 U.S.C. § 3542]. The definition of confidentiality found in FIPS Publication 199 and in NIST SP 800-122 does mention directly that it is concerned with the protection of "personal privacy and proprietary information" [354]. FIPS 199 and NIST SP 800-122 do not define any further technical mechanisms' details let alone their strengths.

NIST SP 800-12 especially mentions that data anonymization for PII confidentiality can be done by subsequent changes or deletion of parts. The definition of actions to increase the confidentiality of PII found in NIST SP 800-122 does mention directly that "[...] [d]eleting an entire record or certain parts of records [...]" [332] can be beneficial, as well as subsequent edits.

9.2.4. Cryptographic definitions of confidentiality based on indistinguishability

Finally, this section provides the background on the confidentiality notion based on indistinguishability that was used in cryptographic papers analysed prior. The idea is to capture the goal of confidentiality in the presence of a defined adversary. In their work from 1984 *Goldwasser and Micali* proposed that encryption, a mechanism to protect confidentiality, shall offer a protection level such that:

> "Whatever is efficiently computable about the cleartext given the cyphertext, is also efficiently computable without the cyphertext."[591] [210]

They also note that their notion is thus "[...] based on complexity theory. Thus, when we say that it is "impossible" for an adversary to compute any information about the cleartext from the cyphertext we mean that it is not computationally feasible."[591] [210] A high-level definition of confidentiality under an efficient, and thus assumed to be polynomially bounded, adversary could be worded as follows:

Definition 134 : Cryptographic Confidentiality (high-level)

> *"[E]xtracting any information about the cleartext from the cyphertext is hard on the average for an adversary with polynomially bounded computational resources"[591] [210]*

This can be also worded as follows: "No polynomially bounded adversary can with non negligible probability distinguish two ciphertexts."[592] The latter wording explains why this property is known as the indistinguishability property.

[591] The American English spelling from original work is retained for terms and quotes.

[592] Posted on `http://crypto.stackexchange.com/questions/20748/proving-the-semantic-security-of-the-one-time-pad` [last accessed: Oct. 2016].

The indistinguishability property has been formally defined for encryption schemes. In the remainder, this section follows the discussion and descriptions given in [42]. The following notion is used: The message space is denoted as M, which is also parameterised by the security parameter λ. Then a public key encryption (PKE) scheme consists of three probabilistic polynomial time (PPT)[593] bounded algorithms KeyGen_{PKE}, Enc, Dec. The key generation algorithm $\mathsf{KeyGen}_{PKE}(1^\lambda)$ outputs a public key pk_{PKE} and a secret key sk_{PKE}. The encryption algorithm takes the public key pk_{PKE} and a message $m \in M$ as inputs and outputs a ciphertext c: $c \leftarrow \mathsf{Enc}(\mathsf{pk}_{PKE}, m)$. The decryption algorithm takes the secret key pk_{PKE} and a ciphertext c as inputs, and outputs a message m: $m \leftarrow \mathsf{Dec}(\mathsf{sk}_{PKE}, c)$.
For correctness of the PKE it is required that:

$$\forall m \in M, \forall(\mathsf{pk}_{PKE}, \mathsf{sk}_{PKE}) \leftarrow \mathsf{KeyGen}_{PKE}(1^\lambda), \forall c \leftarrow \mathsf{Enc}(\mathsf{pk}_{PKE}, m) : \mathsf{Dec}(\mathsf{sk}_{PKE}, c) = m\,.$$

Without loss of generality or any prejudice this thesis chose indistinguishability that is secure against an adaptive chosen-ciphertext attack (IND-CCA2), namely IND-CCA-SE following [41, 42]. In Definition 135 the formalisation following [42] is given in the notation used for the security properties in this thesis.

Definition 135 : CCA2-Indistinguishability

A public-key encryption (PKE) scheme is indistinguishable against an adaptive chosen-ciphertext attack (IND-CCA2) if for any efficient algorithm $\mathcal{A}$ the probability that the experiment $\mathsf{IND\text{-}CCA2}_{\mathcal{A}}(\lambda)$ given in Fig. 45 returns 1 *is negligibly close to $\frac{1}{2}$ (as a function of λ).*

```
Experiment $\mathsf{IND} - \mathsf{CCA2}_{\mathcal{A}}(\lambda)$
  $b \leftarrow \{0,1\}$
  $(\mathsf{pk}_{PKE}, \mathsf{sk}_{PKE}) \leftarrow \mathsf{KeyGen}_{PKE}(1^\lambda)$
  $(m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{Dec1}(\mathsf{sk}_{PKE}, \cdot)}(\mathsf{pk}_{PKE})$
      where oracle Dec1 for input for input $c$
          returns $m \leftarrow \mathsf{Dec}(\mathsf{sk}_{PKE}, c)$.
  // $m_0$ and $m_1$ need both be $\in M$ and of same length
  // to prohibit detections due to failures in the encryption algorithm
  // or due to different lengths of $c^*$.
  $c^* \leftarrow \mathsf{Enc}(\mathsf{pk}_{PKE}, m_b)$
  $a \leftarrow \mathcal{A}^{\mathsf{Dec2}(\mathsf{sk}_{PKE}, \cdot)}(\mathsf{pk}_{PKE})$
      where oracle Dec2 for input for input $c$
          returns $m \leftarrow \mathsf{Dec}(\mathsf{sk}_{PKE}, c)$ iff $c \neq c^*$ and $\perp$ otherwise.
  return 1 if $a = b$, and 0 otherwise
```

Figure 45. Indistinguishability (IND-CCA2) Experiment following [42]

As shown in Fig. 45 by public access to the encryption mechanism the adversary can generate ciphertexts for messages of its choice. It can further check the decryption for chosen ciphertext[594] by having access to a decryption oracle ($\mathsf{Dec1}$). The challenge of the adversary is to identify from two chosen input messages which one gets used to generate the encrypted c^*. After receiving the challenge, the adversary can use the decryption mechanism again — still as a black box — and thus the adversary has access to more plaintexts for chosen ciphertexts. However, the adversary will not be given the plain text for the challenge ciphertext denoted c^*. That is the difference between the first oracle and the second one ($\mathsf{Dec2}$). Note, the above explicitly models in the experiment the inability of the adversary to challenge the decryption oracle in the second round for the challenge ciphertext c^*. Thus the above is IND-CCA-SP following [42], where the 'SP' stands for a penalty (P), i.e., do not allow the adversary within the experiment to query the decryption oracle in the second (S) round[595].

[593] PPT as defined in Definition 11 in Sec. 2.6.

[594] 'CC' in 'CCA' stands for chosen ciphertext.

[595] "IND-CCA-SP implies IND-CCA-SE" [42]. IND-CCA-SE means that the definition does "[...] not have the experiment impose a penalty but just say, outside of the experiment, that the adversary is "not allowed" or just "may not" make a challenge decryption query [...]" [42].

IND-CCA2 describes that an efficient (PPT) adversary can not extract any information about the cleartext from a given ciphertext. In the indistinguishability experiment the adversary can supply two plaintexts and is presented with one ciphertext for which it needs to state which of the two inputs corresponds. If it is indistinguishable, then the encryption is hard to reverse and considered secure.[a]

As indicated the IND-CCA2 definition can be matched to the formal definitions of privacy given in the MSS literature by *Miyazaki et al.*, *Johnson et al.*, and especially the formalisation of the privacy property from *Ateniese et al.* by *Brzuska et al.*. Thus, the cryptographic definition of IND-CCA2 also serves as a model for the privacy definitions to precisely model the adversary's strength and thus the strength of the property. This has been used in the formal privacy definitions of *Miyazaki et al.*[b], *Johnson et al.*[c] and *Brzuska et al.*[d] which define the adversary as a computationally, namely probabilistic polynomial time (PPT)[e] bounded entity.

This observation also added to the formulation of Analysis Result 20[f] (see Sec. 9.4.1).

[a] The intuition behind this is that if the adversary would be able to distinguish with a non-negligible advantage over plain guessing it would indicate that there is an error in the protection as the encrypted output would leak information about its input, which could potentially help to reconstruct it.

[b] "[...] $\mathcal{A}$ be a probabilistic polynomial-time Turing machine." [349].

[c] "[...] every adversary A making at most q chosen-message queries and running in time at most t [...]" [273].

[d] "[...] since A runs in polynomial-time [...]" [64].

[e] Probabilistic polynomial time (PPT) has been defined in Definition 11 in Sec. 2.6.

[f] Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. (see Sec. 9.4.1 on page 233).

9.3. Confidentiality protection levels in technical definition for redaction and data protection

There are existing technical definitions for data protection in general standards, like ISO 29100, but also more specialised ones, e.g., ISO 27038 that are explicitly dealing with the redaction[596] of information. In the following the standards ISO 29100, ISO 27038, ISO 27040, ISO 27042, and ISO 27050, as well NIST Special Publication 800-88 are analysed to identify if they define a minimal required strength of the protection mechanism and if they contain further hints on the technical enforcement through the modifications operations of redaction or sanitization.

9.3.1. ISO 29100 on a framework for privacy

The ISO 29100 version from 2011 is analysed as it "[...] provides a high-level framework for the protection of personally identifiable information (PII) within information and communication technology (ICT) systems. It is general in nature [...]"[597] [250]. It contains the specification of entities and actors and "specifies a common privacy terminology" [250] as well as "provides references to known privacy principles for information technology." [250]. Due to the focus, this thesis is not interested in the reasons for information being considered as personally identifiable information (PII) or even as sensitive PII and the identification processes. The standard does defines it as follows:

> "personally identifiable information
> PII
> any information that (a) can be used to identify the PII principal to whom such information relates, or (b) is or might be directly or indirectly linked to a PII principal
> NOTE To determine whether a PII principal is identifiable, account should be taken of all the means which can reasonably be used by the privacy stakeholder holding the data, or by any other party, to identify that natural person." [250].

[596] Redaction gets defined as the "[...] permanent removal of information within a document [...]" [256] in ISO 27038.

[597] The American English spelling from original work is retained for terms and quotes.

This is in line[598] with the definition of personal data used in this thesis, which comes from the EU's General Data Protection Regulation (GDPR); it is restated from Sec. 8.4.1 for readability.

Definition 125 : Personally identifying information (PII) / personal data from GDPR

> *" 'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person;" [Art. 4 (1) GDPR]*

The ISO standard 29100 also contains a set of privacy principles; a "[...] set of shared values governing the privacy protection of personally identifiable information (PII) when processed in information and communication technology systems [...]" [250]. This set contains eleven privacy principles; three more compared to the eight principles of OECD (see Sec. 8.2.6). The principles of consent and data minimisation have been added as Tab. 8 with the comparison highlights. Both principles can also be found as core concepts in EU GDPR. As the third addition the standard enlists "Privacy compliance" [250] which requires to adhere to legal rules and requirements for data protection. Namely, one must cooperate with legal data protection and other supervisory authorities by "[...] observing their guidelines and [by answering their] requests [...]" [250], and one must provide proofs of compliance "[...] by periodically conducting audits using internal auditors or trusted third party auditors [...]" [250].

ISO 29100 principle	OECD principle	notes
Consent and choice	-	in Article 4 (11) GDPR (Sec. 2.16)
Purpose legitimacy and specification	Purpose Specification	
Collection limitation	Collection Limitation	
Data minimization	-	in Article 5 (c) GDPR (Sec. 8.2.7)
Use, retention and disclosure limitation	Use Limitation	
Accuracy and quality	Data Quality	
Openness, transparency and notice	Openness	
Individual participation and access	Individual Participation	
Accountability	Accountability	
Information security	Security Safeguards	
Privacy compliance	-	

Table 8. Mapping the titles of the 11 privacy principles from ISO 29100 and those from OECD discussed in Sec. 8.2.6

Regarding the data minimisation, which can also be found in Article 5 (c) GDPR (see Sec. 8.2.7), the ISO 29100 sees this as "[...] closely linked to the principle of "collection limitation" but [data minimisation] goes further than that." [250]. Especially, ISO 29100 highlights that the design and implementation information and communication technology system shall enforce "[...] a "need-to-know" principle, i.e. one should be given access only to the PII which is necessary for the conduct of his/her official duties in the framework of the legitimate purpose of the PII processing;" [250]. One way to technically reach privacy protection are so called 'privacy-enhancing technologies', or short PET; the 2011 version of ISO 29100 defines them as follows:

> "privacy-enhancing technology
> PET
> privacy control, consisting of information and communication technology (ICT) measures, products, or services that protect privacy by eliminating or reducing personally identifiable information (PII) or by preventing unnecessary and/or undesired processing of PII, all without losing the functionality of the ICT system

[598] While in direct textual comparison there might be differences, namely from the excerpts presented above there is the additional note in ISO 29100 that "all the means which can reasonably be used by the privacy stakeholder holding the data, or by any other party, to identify that natural person" [250] have to be taken into account. This might get wrongly interpreted as reading stricter than GDPR. However, it is in line with the GDPR, which states – additionally to Art. 4 (1) – in Recital 26 that in order "[t]o determine whether a natural person is identifiable, account should be taken of all the means reasonably likely to be used, such as singling out, either by the controller or by another person to identify the natural person directly or indirectly. [...]" [Recital 61 GDPR].

> NOTE Examples of PETs include, but are not limited to, anonymization and pseudonymization tools that **eliminate**, reduce, **mask**, or **de-identify** PII or that prevent unnecessary, unauthorized and/or undesirable processing of PII.
> NOTE Masking is the process of **obscuring elements** of PII."[599] [250].

The above mentioned notion of a "privacy control" [250] is defined as "measures that treat privacy risks by reducing their likelihood or their consequences" [250]. Among the operations carried out by PETs ISO 29100 lists the elimination or masking, both are done in order to remove or obscure blocks from a set of information that is regarded as personal data.

ISO 29100 lists operations like masking and removal for PETs in order to prohibit access to unnecessary personal information by withholding it. ISO 29100 contains the privacy principle of data minimisation. This would indeed require to prohibit the access to unnecessary personal information by withholding it. Namely, ISO 29100 proposes to "[...] delete and dispose of PII whenever the purpose for PII processing has expired [...]" [250]. Redaction or sanitization that provide confidentiality for removed or over-written blocks would provide such data minimisation. While ISO 29100 does not contain the terms 'redaction' nor 'sanitization', the text regarding the privacy-enhancing technologies (PET) contains similarly worded operations: "[...] terminate, reduce, mask, or de-identify PII [...]" [250]. Quite explicitly masking gets defined as "[...] the process of obscuring elements of PII." [250]. ISO 29100 does describe that blocks ("[...] elements of PII." [250]) can get masked or eliminated. Elimination can be done when redaction removes the information with strong enough cryptographic privacy guarantees. Searching the ISO standards[a] for the notion of 'obscuring' the term "obscure" [259] in IT context is defined in ISO 2832 in the version from 2015 as meaning to "[...] block the user's view of part or all of a displayed object with another [...]" [259].

The ISO 29100 standard does not make statements regarding the exact mechanisms nor their required strength. This was also not expected as it is a framework standard. The framework demands that a privacy risk assessment (PIA) to be carried out in order to identify the organisation's individual privacy safeguarding requirements, which then identifies the "[...] appropriate controls at the operational, functional and strategic level to ensure the integrity, confidentiality and availability of the PII, and protect it against risks such as unauthorized access, destruction, use, modification, disclosure or loss throughout the whole of its life cycle [...]" [250, p. 18].

[a] ISO offers the Online Browsing Platform (OBP) at `https://www.iso.org/obp/ui/` [last accessed Feb. 2017].

9.3.2. ISO 27042 on digital evidence mentions sanitization and redaction

ISO 27042 is set forth to provide "[...] guidance on the conduct of the analysis and interpretation of potential digital evidence in order to identify and evaluate digital evidence which can be used to aid understanding of an incident [...]" [258]. Without going into the details of this ISO standard the analysis done for this thesis found that the terms 'sanitization' and 'redaction' are mentioned in the introduction giving relations to two further standards (ISO 27038 and ISO 27040). They are as follows:

> "Some documents can contain information that must not be disclosed to some communities. Modified documents can be released to these communities after an appropriate processing of the original document. The process of removing information that is not to be disclosed is called "redaction". The digital **redaction** of documents is a relatively new area of document management practice, raising unique issues and potential risks. Where digital documents are redacted, **removed information must not be recoverable**. Hence, care needs to be taken so that **redacted information is permanently removed** from the digital document (e.g. it must not be simply hidden within non-displayable portions of the document). ISO/IEC 27038 specifies methods for digital redaction of digital documents." [258]

> "[...] [ISO/IEC 27040:2015] provides detailed technical guidance on how organizations can define an appropriate level of risk mitigation by employing a well-proven and consistent approach to the planning, design, documentation, and implementation of data storage security. [...] Security mechanisms like encryption and **sanitization** can affect one's ability

[599] **Bold** face and other *emphasis* like ~~deletions~~ or underlining have been added for highlighting.

> to investigate by introducing obfuscation mechanisms. They have to be considered prior to and during the conduct of an investigation. They can also be important in ensuring that storage of evidential material during and after an investigation is adequately prepared and secured."[600] [258]

ISO 27042 concurs with the need for modifications by redaction and sanitization, but raises concerns over it being potential problem for digital evidence. The ISO standard acknowledges that there are situations when "[...] information [...] must not be disclosed to some communities [...]" [258]. With this it is in line with Analysis Result 19[a]. On the one hand it states, pointing out a relation to ISO 27038[b], that after "redaction" [258] care must be taken to ensure that the "[...] redacted information is permanently removed from the digital document [...]" [258]. As the analysis in Sec. 9.3.5 will show, ISO-27038 does not discuss the redaction or sanitization of integrity or authenticity-protected digital documents. On the other hand ISO 27042 clearly highlights a potential negative impact of the subsequent modifications: They "[...] can affect one's ability to investigate by introducing obfuscation mechanisms." [258]. This thesis with the analysis of the probative value of MSS can aid to establish MSS as a suitable tool to solve a digital evidence sanitization problem.

[a] Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism (see Sec. 8.4.6 on page 211).

[b] ISO 27038 has the title 'Specification for digital redaction'. [256]

The two above mentioned ISO standards and related standards are analysed in more detail in the subsections to follow (ISO 27040 in Sec. 9.3.3 and ISO 27038 in Sec. 9.3.5).

9.3.3. Definition of sanitize / sanitization from ISO 27040 and ISO 27050

Both ISO standards, ISO 27040 [257] on storage security and ISO 27050 [261] on electronic discovery, describe the process of 'sanitization' or the action to 'sanitize'. ISO 27050 is especially interesting as it defines this term in the context of removing information from data which might be forwarded to another party during the process of electronic discovery. Electronic discovery "[...] is the process of discovering pertinent Electronically Stored Information (ESI) or data by one or more parties involved in an investigation or litigation, or similar proceeding [...]" [261]. The following definitions are given for terminology related to sanitization:

Definition 136 : Sanitize from ISO 27050

> *"**sanitize***
> *process to remove information from media such that data recovery is not possible at a given level of effort*
> *[SOURCE: ISO/IEC 27040:2015, 3.38, modified]*
> *Note 1 to entry: Clear, purge, and destruct are actions that can be taken to sanitize storage media."*[600] *[261]*

Definition 137 : Sanitize / sanitization from ISO 27040

> *"3.37*
> *sanitization*
> *process or method to sanitize (3.38)*
>
> *3.38*
> *sanitize*
> *render access to target data (3.52) on storage media (3.48) infeasible for a given level of effort*
> *[...]"*[600] *[257]*

ISO 27040 and ISO 27050 define the goal of sanitization to make data inaccessible for a given level of effort by the adversary. ISO 27040 describes the goal of sanitization as to "[...] render access [...] infeasible [...]" [257]. In line with this ISO 27050 requires that the data is removed "[...] such that data recovery is not possible [...]" [261]. The notions are in line with the understanding of

[600] The American English spelling from original work is retained for terms and quotes.

a valid sanitization used in this thesis, which was given in Definition 47 on page 49. Both standard's definitions require by the phrase "[...] a given level of effort [...]" [257, 261] to state that the strength of the adversary which the actual mechanism can withstand is not within the standard's scope. Hence, the legally compliant strength must be defined externally and must be checked for each mechanism deployed.

There are then additional technical documents that describe how the mechanisms shall work in order to achieve certain levels of security: Examples are NIST SP 800-88 [283] which is concerned with media sanitization or the DIN 66399-2 [155] that describes the requirements for equipment for destruction of data carriers for the standard. The former is selected for further analysis due to its availability and it being mentioned in analysed legal texts.

9.3.4. Sanitization of electronic media following NIST SP 800-88

Revision 1 from 2014 of the NIST Special Publication 800-88 is concerned with media sanitization; by this term NIST SP 800-88 "[...] refers to a process that renders access to target data on the media infeasible for a given level of effort [...]" [283, p. iii]. It was being mentioned in the context of US law, namely U.S.R. Title 45 - § 300jj-12, which was analysed in Sec. 8.2.2. NIST SP 800-88 includes requirements on the physical destruction of physical storage media[601] as well as logical methods that "[...] best mitigate the risks to an unauthorized disclosure of information [...]" [283, p. 24]. For example "[...] to overwrite user-addressable storage space on the media with non-sensitive data, using the standard read and write commands for the device [...]" [283, p. 24].

NIST SP 800-88 contains no new insights regarding the confidentiality protection of blocks of messages by redaction or sanitization. The sole focus is the deletion of data from storage media including the physical destruction of the storage medium. It follows the goal that "[t]he organization sanitizes or destroys information system digital media before its disposal or release for reuse outside the organization, to prevent unauthorized individuals from gaining access to and using the information contained on the media [...]"[a] [283, p. 1].

Even though NIST SP 800-88 got mentioned in US law it does not offer further insight regarding the kind of sanitization and redaction that MSS offer.

[a] The American English spelling from original work is retained for terms and quotes.

9.3.5. Redaction of digital documents following ISO 27038

The ISO standard 27038 is dedicated to the redaction of digital documents. It describes the effects and implicitly the requirements of the redaction process as follows:

Definition 138 : Redaction from ISO 27038

> *"2.4*
> ***redaction***
> *permanent removal of information within a document" [256]*

In more detail, when listing the goals, ISO 27038 uses the phrases "securely removed" [256] or "permanent removal" [256] and states that a "[r]edaction shall irreversibly remove the required information from the redacted version of the document [...]" [256].

The standard further distinguishes two forms of redaction: basic and enhanced.

Definition 139 : Basic redaction from ISO 27038

> *"**6.6.2 Basic redaction***
>
> *Basic redaction shall result in the permanent removal of the required information. Redactions may be marked in any way – for example by the replacement of the redacted information with 'blacked-out' symbols replacing each redacted character." [256, p. 6]*

[601] "Use optical disk media shredders or disintegrator devices to reduce to particles that have a nominal edge dimensions of 0.5 mm and surface area of 0.25 mm^2 or smaller [...]" [283, p. 36].

Definition 140 : Enhanced redaction from ISO 27038

> *"6.6.3 Enhanced redaction*
>
> *Enhanced redaction shall result in the permanent removal of the required information, along with such contextual information that could be used to identify the redacted information. All redacted information within a particular document shall be marked in the same manner. The length and other appropriate attributes of the redaction markings shall be consistent within a particular document." [256, p. 6]*

The notion of a 'basic redaction' from ISO 27038 allows, but does not mandate[602], to mark where redactions have been. Also the 'enhanced redaction' speaks of redactions getting marked. Herein the standard even uses the keyword 'shall', which in ISO standards indicates a requirement[603]. This is interpreted that a basic might and an enhanced redaction following ISO 27038 must result in a redacted document that allows the Verifier to detect that a redaction has taken place. Transferred to the privacy properties of MSS and the new extended integrity notion this was termed a non-interactively publicly accountable RSS (see Sec. 13.1.2).

ISO 27038 describes in natural high-level language the goals and strength of a technical redaction process and excludes redactions that cannot be detected, i.e., enhanced redactions cannot be transparent. Regarding the scope, authenticity and integrity-protected digital documents can be subject to ISO 27038 compliant redactions. While ISO 27038 does explicitly exclude information from databases, it does not exclude authenticity-protected digital documents. However, ISO 27038 does not include any statement what would or should happen to the verifiability of authenticity and integrity for the redacted version.

Especially, ISO 27038 does not negate Analysis Result 11[a].

ISO 27038 requires ("should") for enhanced redactions and recommends ("may") for basic redactions the public form of accountability (PUB), namely non-interactively publicly accountable. In the standard's general principles of digital redaction this is mentioned as "[t]he process of digital redaction is not simply to remove information but also to indicate where necessary that some information has been removed [...]" [256, p. 2]. At least the "[...] reader [in MSS the Verifier] knows that the document has been redacted [...]" [256, p. 2]. The standards gives as reason to "[...] maintain the semantics of the non-redacted information [...]" [256, p. 2], which is in line with the examples given in Sec. 3.10 when introducing and discussing the notion of transparency on a high-level. This is in line with Analysis Result 10[b].

Regarding the strength of the redaction ISO 27038 clearly notes that "[r]edaction is carried out in order to permanently remove particular information [...]" [256, p. 2]. Even stronger than this goal, ISO 27038's section on redaction principles states the requirement[c] "[r]edaction shall irreversibly remove the required information from the redacted version of the document [...]" [256, p. 3].

[a] Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect (see Sec. 5.5.4).

[b] Analysis Result 10: Detection of unauthorized modifications is strictly required; it might even be required for any subsequent modification (see Sec. 5.5.3 on page 136).

[c] The keyword 'shall' indicates a requirement in ISO standards (cf. footnote 603).

9.4. Analysis results of technical definitions regarding confidentiality protection of personal data and trade secrets

To conclude, this section compiles in condensed and summarised form the intermediate findings and results obtained from this chapter's analysis of the computer science related texts. The following four analysis results, together with the previous analysis results of legal texts, form the basis for the list of technical requirements towards the sanitization[604] (including redaction) of data for a legal data protection. These requirements are presented in Chapter 10 of this thesis.

[602] Uses the not so strong 'may' in "Redactions may be marked in any way" [256]. The notion of 'may' is used in ISO standards to indicate that something is permitted.

[603] See among other sources `https://www.iso.org/foreword-supplementary-information.html` [last accessed: Jan. 2017].

[604] The American English spelling from original work is retained for terms and quotes.

9.4.1. Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy.

The analysis of different privacy definitions from influential earlier works in the field of RSS and SSS (see Sec. 9.1) found that they all agree on a common underlying notion of the non reconstructability for the original, i.e., not redacted or not sanitized, information. While the analysis also showed differences in detail, the modelling of this core property of MSS based on the indistinguishability notion defined for encryption schemes allows to define the strength of cryptographic[605] privacy. In the definitions of the previously existing and the more recent newer definitions, the adversary is a computationally, namely polynomial-time (PPT), bounded entity. In this regard, ISO 27040 and ISO 27050 explicitly define the goal of sanitization as to make data inaccessible for a given level of effort by the adversary.

However, in the same way that there are "[s]ubtleties in the [d]efinition" [42] of cryptographic indistinguishability (cf. [42]) there are several differentiating factors in the privacy model regarding the adversary's capabilities, i.e., access to oracles or control over key pairs. Those differences result in privacy definitions with different cryptographic strength: As a starting point this thesis selects the privacy property originally described by *Ateniese et al.* in the formalisation of *Brzuska et al.*. This property was defined for SSS. As this is the baseline, this notion of privacy is termed standard privacy. Following the works of *Brzuska et al.* [66] this was also used to define privacy for RSS. The definition is presented next as Analysis Result 21. The stronger notions include the strengthened privacy[606] as presented in Sec. 15.1.3, the property of transparency as defined also formally by *Brzuska et al.* and presented in Sec. 11.6.4 and Sec. 11.13.4, as well as the property of unlinkability in Sec. 13.3.2 that for instance can be constructed with group signatures as given by *Brzuska et al.* [67].

9.4.2. Analysis Result 21: The definition of the cryptographic property of standard privacy

The analysis of several existing formal and informal definitions in the realm of MSS yielded that the definition by *Brzuska et al.* given in [64] describe in sufficient detail existing properties or allows to understand subtle differences to existing definitions. The definition from [64] is widely known in the MSS community: Looking at the existing body of work, it was cited and thus it was carried forward into numerous works in the domain of RSS and SSS, e.g., [23, 31, 275, 294, 305, 341, 342, 358, 443, 444, 499, 509, 510] and others[607]. The definition from [64] already saw scrutiny, e.g., *Gong et al.* rectified a weakness for *Brzuska et al.*'s security model in [216]. Hence, it was selected as the *standard* privacy notion for MSS and is defined as follows:

Definition 156 : Standard Privacy for SSS following *Brzuska et al.*

> *An SSS is **private**, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 51 returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ).*

For details please see the discussion offered in Sec. 11.6.3 from which the above Definition 156 and Fig. 51 on page 234 for SSS are re-stated. For RSS the respective definition is Definition 174 with Fig. 60 on page 289.

9.4.3. Analysis Result 22: Leakage through the message is out of scope of MSS properties

The analysis of several existing formal and informal definitions in the realm of MSS yielded that definitions defining the information gain an adversary has, e.g., the privacy or transparency definitions, make explicit that any leakage of information through the modified message is out of scope of MSS properties. See for example the following statement by *Brzuska et al.*:

[605] This is added to distinguish it from the legal use of the term.

[606] It gives the adversary access to the Sanitizer's key.

[607] This is by no means meant as an exhaustive list and it does not include the citations for *Brzuska et al.*'s work in the area of RSS, e.g., [66].

```
**Experiment** $\mathsf{Privacy}^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}) \leftarrow \mathsf{KGen_{sig}}(1^{\lambda})$
  $(\mathsf{pk_{san}}, \mathsf{pk_{san}}) \leftarrow \mathsf{KGen_{san}}(1^{\lambda})$
  $b \overset{\$}{\leftarrow} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^{\lambda}, \mathsf{sk_{sig}}, \cdots), \mathsf{Sanit}(1^{\lambda}, \cdots, \mathsf{sk_{san}})}_{\mathsf{Proof}(1^{\lambda}, \mathsf{sk_{sig}}, \cdots), \mathsf{LoRSanit}(1^{\lambda}, \ldots, \mathsf{sk_{sig}}, \mathsf{sk_{san}}, b)}(1^{\lambda}, \mathsf{pk_{sig}}, \mathsf{pk_{san}})$
    where oracle LoRSanit on input of $m_0, \mathsf{MOD}_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$ // resulting $m'$ is the same
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{ADM} \not\subseteq m_0$ or $\mathsf{ADM} \not\subseteq m_1$ or $\mathsf{MOD}_0(\mathsf{ADM}) \not\subseteq \mathsf{MOD}_0(m_0)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}) \neq \mathsf{MOD}_1(\mathsf{ADM})$, return $\perp$
      let $(m_i, \sigma_i) \leftarrow \mathsf{Sign}(1^{\lambda}, m_b, \mathsf{sk_{sig}}, \mathsf{pk_{san}}, \mathsf{ADM})$
      return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^{\lambda}, m_i, \mathsf{MOD}_b, \sigma, \mathsf{pk_{sig}}, \mathsf{sk_{san}})$
  return 1, if $a = b$
```

Figure 51. Standard Privacy Experiment for SSS based on *Brzuska et al.* [64] including additions from *Krenn et al.* [294]
(same Figure as on page 265)

> "Privacy roughly means that it should be infeasible to recover information about the sanitized parts of the message. As information leakage through the modified message itself can never be prevented, we only refer to information which is available through the sanitized signature." [64]

Also in the classical digital signature schemes one can find the following 'design choice': Mechanisms for digital signature generation shall work regardless of the semantic contents of the data being technically signed. They work on a bitstring and are thus generally applicable to a wide range of applications.

In fact, the definition in this thesis states that the message is a bitstring, i.e., $m \in \{0,1\}^*$. (see Sec. 2.1). In the same way, the analysed technical standards did not make assumption on the semantics of the message. Also the body of cryptographic works analysed defined the message to be from a (maybe length restricted[608]) message space — the use of $\{0,1\}^*$ can be often found[609] — without any further assumptions on the messages' semantics.

EXAMPLE

To give three examples, assume a cryptographically sound subsequent modification of one block and the correct adjustment of the signature does not influence if the block's original content (or even its information) is still semantically contained in the block(s) after the modification. Assume a string in the first block $m_1 =$ "secret" is only modified into $m_1' =$ "hidden,before:secret". Another example, if a three block long message contains $m_1 =$ "secret" and $m_3 =$ "secret" then a redaction of only the first or the last block would not remove the information itself, as it is carried as a semantic duplicate in another block. Finally, if a block gets sanitized from the string "canis lupus" into the string "wolf" then this will only hide the information that the original term was given in latin language, but retain some information assuming the intended recipient understands English. In each example, the occurred information leak is on the semantic level through the message.

When the signature mechanism has no knowledge of the messages's semantic, a security property of the mechanism can not consider it neither. The mechanism has, however, control over the semantics of the signature it generates. Hence, removing information from a signed message is described as prohibiting leakage of the information only through the adjusted signature after the redaction or sanitization.

[608] *Katz and Lindell* define it as "$\{0,1\}^{l(n)}$" with l(n) being the message's length [278].

[609] $\{0,1\}^*$ can be found for example in [12, 64, 278] and many more.

9.4.4. Analysis Result 23: Cryptographic and legal notion of transparency have an opposite meaning

The analysis found a contradiction between the legal concept of 'transparency (legal)' given in Definition 143 with the cryptographic property of 'transparency' that describes a security property of malleable signature schemes as given in Definition 157. The legal notion is close to the general language usage; the term 'Transparency' describes what is visible or accessible and hence has "[...] the quality that makes something obvious or easy to understand [...]"[610] to an observer. Computer scientists have started to use 'transparency' to describe the opposing goal, i.e., describing abstraction in order to hide underlying details. Examples are statements like "Caches are usually transparent or invisible to the processor." [406, p. 85][611], or the principle of transparency in distributed systems. The latter states that transparency is to hide the distributed character of the system to the user or software developer [405, p. 707][612]. The computer science usage is "[...] closer to meaning invisible or undetectable [...]" [464] as stated by online glossaries for computer science terms[613]. This usage is in line with influential[614] works on computer languages: The term can be traced back to *Quine* coining the term "referential transparency" [401] for computer languages. Referential transparency means that one does not need to know how some referenced sub-function works, but just its resulting value following *Strachey*.[615]

9.4.4.1. Cryptographic definition of transparency

The following high level introduction to the cryptographic notion of transparency introduced in the area of MSS by *Ateniese et al.* [12] was already shown in Sec. 3.10.1 to provide this central concept early on. Cryptographically, transparency describes an indistinguishability property that allows that a certain action, e.g., the execution of an algorithm that modified a message, are kept unrecognisably hidden from another party. This notion is especially used in the area of malleable signatures, and informally means the following:

Definition 157 : Cryptographic Transparency

> ***Transparency*** *prevents third parties which have only access to a given valid message-signature pair (m, σ) and public information (e.g., pk_{sig} or pk_{san}) to decide which party (Signer or Sanitizer) is accountable for a given valid message-signature pair (m, σ).*
>
> *Notes for Definition 157:*
>
> *Note 1 Information leakage through the message itself is out of scope of Definition 157.*
>
> *Note 2 Transparency is cryptographically stronger than the cryptographic privacy notion (see Definition 155), i.e., transparency implies privacy [64].*
>
> *Note 3 Third party refers here to any party that is neither Signer or Sanitizer, i.e. an entity in the role of the Verifier. The third party has no access to non-public information; especially not to the secret keys (sk_{sig} or sk_{san}) or output generated involving them, i.e. no access to the proof π for the given valid message-signature pair (m, σ).*

A formal definition is given in Sec. 11.6.4 for SSS and in Sec. 11.13.4 for RSS.

[610] `http://www.learnersdictionary.com/definition/transparency`.

[611] *Reilly* uses the term transparent in the same meaning on the topic of virtual memory: "This is achieved by placing the contents of the large, virtual memory on an auxiliary storage device and bringing parts of it into the main memory, as required by the program, in a way that is transparent to the program." [406, p. 146].

[612] Found in the German 'Handbook' by *Rechenberg and Pomberger*; in the German original text: "Transparenz: Dem Benutzer [...] soll sich das verteilte System als einzige abstrakte (nebenläufige) Maschine präsentieren." [405].

[613] "[...] closer to meaning invisible or undetectable. Computer programs and procedures that are said to be transparent are typically those that the user is – or could be – unaware of." [464].

[614] Appearing as a cite in the 'Fundamental Concepts in Programming Languages' [458] by *Strachey* in the year 1967 is considered as influential.

[615] See also *Søndergaard and Sestoft* [450] for a discussion on the usage of the term.

9.4.4.2. Legal definition of transparency

The notion of transparency is mentioned in many legislations and is an important legal principle: The "[...] openness enables citizens to participate more closely in the decision-making process and guarantees that the administration enjoys greater legitimacy and is more effective and more accountable to the citizen in a democratic system." [167, para 54]. The aforementioned statement was made by the EU Court of Justice in referral to Recital 2 of Regulation No. 1049/200[616] [177]. Furthermore, already the Treaty on European Union[617] highlights transparency for example in Art. 1 Sec. 2 and Art. 11 Sec. 2 [336][618].

According to data protection law transparency denotes that the affected person must be informed, which data is being collected about the person, for which purpose, for how long and which rights the person can exercise. With respect to IT systems transparency is designed to make (IT-) procedures understandable and controllable. This implies the free and easy access to readily available government information, the enactment of swift control and participation procedures, the creation of specialised and independent bodies to control and check [227].

Already previous to the new European General Data Protection Regulation (GDPR), the balance between the right to privacy with respect to the processing of personal data and the principle of transparency when restricting the free flow of personal data was addressed in the Regulation No. 45/2001 [176] and concretised by the European Court of Justice in Case C-28/08 P of 29th June 2010 [167]. Following the now outdated German Federal Data Protection Act (BDSG) which implements Directive 95/46/EC on the protection of individuals with regard to the processing of personal data and on the free movement of such data (Directive 95/46/EC) one can identify what the legal definition of transparency might be.

The GDPR contains regulations regarding "[t]he principle of transparency" [GDPR], e.g., Art. 12 GDPR. It can be distilled to the following definition:

Definition 141 : Principle of Transparency following EU GDPR

> *"The principle of transparency requires that any information [...] be concise, easily accessible and easy to understand [...]"[619] [Recital 58 GDPR].*

This is in line with legal texts prior to the GDPR: For example, following the commentary on BDSG [503] [621] a definition of the principle of transparency can be instantiated from several guarantees given to the data subject, e.g., Right of Information codified in § 4 Abs. 3 BDSG or following from Article 10 Directive 95/46/EC.

Definition 142 : Principle of Transparency inherent in the German BDSG

> *"The data processing should be comprehensible or open to scrutiny by the data subject." [620] [621][503]*

In light of the above, this thesis will summarise the notion as follows:

Definition 143 : Transparency (legal)

> *In the legal context, transparency means openness of procedures.*

[616] In full it reads as follows: "Openness enables citizens to participate more closely in the decision-making process and guarantees that the administration enjoys greater legitimacy and is more effective and more accountable to the citizen in a democratic system. Openness contributes to strengthening the principles of democracy and respect for fundamental rights as laid down in Article 6 of the EU Treaty and in the Charter of Fundamental Rights of the European Union." [177, recital 2].

[617] Note, that this is the highest legal texts in the hierarchy of norms in the EU (see Fig. 10).

[618] Namely, the EU Treaty states that transparency means that "[...] decisions are taken as openly as possible [...]" [336, Art. 1 Sec. 2] and means that the government needs to "[...] maintain an open, transparent and regular dialogue [...]" [336, Art. 11 Sec. 2].

[619] "The principle of transparency requires that any information addressed to the public or to the data subject be concise, easily accessible and easy to understand, and that clear and plain language and, additionally, where appropriate, visualisation be used." [Recital 58 GDPR].

[620] German-English translation according to Langenscheidt Recht Englisch [70].

[621] Recital 43 from *Wolff* [503]'s commentary states in German "[...] Datenverarbeitung insgesamt muss für den Betroffenen **möglichst nachvollziehbar** sein [...]" [503].

9.4.4.3. Legal transparency vs. cryptographic transparency

A cryptographically transparent modification of signed data renders the occurred modification unrecognisably hidden for the Verifier, according to the Definition 157 on page 266. This contradicts first and foremost the legal transparency according to the previously proclaimed Definition 142 on page 236. Further, cryptographic transparency comes amiss the legal requirement to detect any subsequent modification (authorized or unauthorized) set forth by the Directive 1999/93/EC and Regulation 910/2014 (see also Analysis Result 4[622]) — at least for a Verifier this can not be achieved without further interactions. The cryptographic property of accountability enables cryptographically transparent signature schemes to move a step closer toward legal transparency by offering to identify the party accountable, but this might require help by a party with access to secret keys. More discussions on the link between accountability and detectability can be found for example in Sec. 6.4 and in Sec. 17.3.4.

The legal and cryptographic notion of transparency differ significantly. The alignment between cryptographic primitives and legal requirements is initially hindered by the fact that the term ‘transparency’ has a different, if not even an opposite meaning in the field of legal texts and cryptographic research.

To avoid further confusion but to not oppose current cryptographic literature this thesis will keep them both, but differentiates:

- *Transparency (legal)* refers to transparency in the legal context.
- *Transparency* denotes the cryptographic property following the terminology introduced in the existing literature of for malleable signature schemes, e.g., by *Ateniese et al.* [12] for sanitizable signature schemes (SSS).

[622] Analysis Result 4: Directive 1999/93/EC and Regulation 910/2014 require that any subsequent modification must be detected (see Sec. 4.6.4 on page 105).

10 Technical requirements for legally compliant privacy protection through sanitization

Overview of Chapter 10

This chapter concludes Part II [623] with the formulation of the last two requirements: Requirement 9[624] and Requirement 10[625]. The presentation of these two last technical requirements concludes the steps from the methodology to formulate technically precise requirements for a legally compliant removal of information through sanitization (including redactions)[626]. Both requirements contain the specific properties that are demanded by the first part of the research question. To recall, the research question is:

> Can a malleable signature scheme be private to be compliant with EU data protection regulation **and at the same time** fulfil the integrity protection legally required in the EU to achieve a high probative value for the data signed?

In this section the use of the key word MUST (or MUST NOT) indicates a mandatory property that is needed in order to offer a sophisticated level of confidentiality. An optional property is indicated by MAY (or MAY NOT). This use of keywords follows RFC 2119 [62]. It concentrates on MSS functionality, which is in line with Analysis Result 19[627] that highlighted the legal suitability of technical mechanism of sanitizations (including redactions).

Sec. 10.1 states the mandatory Requirement 9, which requires the mechanism MUST prohibit a recovery of information after the information was successfully sanitized or redacted with a legally accepted state-of-practice strength. Namely, Requirement 9 contains a technical requirement regarding the confidentiality level for legal compliance with EU data protection. It is formulated such that it can be described through the cryptographic notion of privacy as defined for SSS and RSS.

Sec. 10.2 states the recommended Requirement 10. which explicitly states that the confidentiality protection protection MAY be further strengthened beyond Requirement 9 but not conflict with the previous 8 requirements on the integrity protection legally required for an increased probative value.

[623] Note, Part II is structured in the same way as Part I to ease readability of the thesis.

[624] See Sec. 10.1.

[625] See Sec. 10.2.

[626] Step 2.1 (Define a precise enough terminology to capture the requirement) was easier for privacy than for integrity as no new terminology was needed. This chapter contains the result of step 2.2. (Formalise application requirements (including legal) using technically precise terminology) from the methodology given in Sec. 1.5.

[627] Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism (see Sec. 8.4.6 on page 211).

10.1. Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength)

This ninth requirement captures as technically precise as possible the needed protection level against reconstruction of the original data that is offered by sanitization and redactions to become legally compliant. If this occurs, this is termed a *leakage*[628]. Analysis Result 19[629] identified the legal suitability of the technical mechanism of sanitization and redaction. Moreover, the definition of personal data given as Analysis Result 14[630]. Any compliant technical mechanism's strength must fulfil the strongly demanded protection of personal data that was stated as Analysis Result 16[631]. MSS offer data minimisation capabilities, as they allow the sharing of only selected — still authentic — block(s) with authorized entities. Finally, the indication that it is legally allowed to remove data for trade secret protection as well, given as Analysis Result 15[632], does not contradict the suitability of this mechanism. The analysis of technical texts showed that the needed protection goal is close to that of confidentiality, which was captured as Analysis Result 20[633]. Further, analysis yielded a technically precise formulation of the cryptographic property given as Analysis Result 21[634].

Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength)

An integrity protection mechanism that allows authorized subsequent changes of blocks (ACA-integrity protection) MUST *offer reliable protection against recovery of information from the block(s) within the integrity-protected message that were overwritten or removed by a valid authorized subsequent modification. Reliably means that the mechanism* MUST *be*

1. *considered cryptographically secure against state-of-the-art attacks, and*
2. *that the algorithm's strength must be legally compliant.*

Note 1 Only leakage of information about the original content of a block, which was overwritten by a valid sanitization or removed by a valid redaction, through the signature is considered.

Note 2 The ability of accountability MUST NOT *be affected by achieving this kind of confidentiality.*

Note 1 follows from the Analysis Result 22[635]; see Sec. 9.4.3 on page 233 for more details and an example. Note 2 explicitly states what was shown in the work of *Brzuska et al.*: They showed the "[i]ndependence of Signer-Accountability" [64] as well as the "[i]ndependence of Sanitizer-Accountability" [64]. Thus, note 2 makes this explicit in this requirement, as previous Analysis Result 12[636] has demanded accountability.

Requirement 9 contains a cryptographic and a legal component. The thesis highlights this by splitting the requirement into two sub-requirements. Requirement 9.1 selects the most suitable cryptographic definition for the notion of protection against reconstruction including the demand to allow for a specification of its cryptographic strength. After that, Requirement 9.2 deduces the legally suitable strength from what the legal texts state directly or indirectly through technical definitions.

[628] Following the wording of works like [300] with the title: "Leakage-Free Redactable Signatures".

[629] Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism (see Sec. 8.4.6 on page 211).

[630] Analysis Result 14: Definition of personal data (in Sec. 8.4.1 on page 207).

[631] Analysis Result 16: Legal texts acknowledge the importance of confidentiality protection of personal data and the high-level descriptions of the goal of confidentiality coincide (see Sec. 8.4.3 on page 208).

[632] Analysis Result 15: Definition of trade secret (in Sec. 8.4.2 on page 207).

[633] Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. (see Sec. 9.4.1 on page 233).

[634] Analysis Result 21: The definition of the cryptographic property of standard privacy (see Sec. 9.4.2 on page 233).

[635] Analysis Result 22: Leakage through the message is out of scope of MSS properties (see Sec. 9.4.3 on page 233).

[636] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

10.1.1. Requirement 9.1 (Mechanism MUST achieve a suitable cryptographic protection against the recovery of sanitized or redacted data from the adjusted signature)

The protection goal to achieve by the MSS[637] is to ensure that the authorized subsequent modifications of integrity-protected data indeed sanitize or remove the trade secret or personal information contained in the original data. This needs to happen such that the original data are no longer reconstructable by the adversary from the adjusted signature.

Following the original body of work from MSS the cryptographic property that captures Requirement 9 is termed *privacy*[638]. The author of this thesis is well aware that the RFC 4949 recommends not to use ‘privacy’ to describe a data confidentiality related security property. More specifically, RFC 4949 states that “[p]rivacy is a reason for security rather than a kind of security.” [438] (see Sec. 9.2.2). However, this is the formerly coined name of that property.

In this thesis the MSS security property of ‘cryptographic privacy’ describes the data confidentiality protection against recovery of sanitized or redacted original data. Contrary to RFC 4949 the formerly coined term ‘privacy’ is kept to stay consistent with existing literature in the realm of MSS. In particular, after the formalisation of the property by *Brzuska et al.*’s in [64–66] the term has been picked up in the MSS literature to describe the confidentiality offering of MSS[a]. To not confuse expert readers this central term was not changed.

[a] See Sec. 9.2.2 and the works by *Slamanig and Rass* [441], *Canard et al.* [112], *Attrapadung et al.* [15] , or *Kundu and Bertino* [299] are examples of the term being broadly used in MSS literature.

The property of cryptographic privacy can be defined analog to the indistinguishability of encryption schemes, as brought up in Analysis Result 20[639]. A suitable formalisation of this central cryptographic property, termed standard privacy, was thus formulated as Analysis Result 21[640].

10.1.2. Requirement 9.2 (Mechanism MUST be as good as the legally accepted current state of practice for protection against unauthorized recovery of sanitized or redacted information)

The privacy property of the MSS shall achieve a level of protection against unauthorized recovery that allows for complying with EU data protection. In general, Analysis Result 19[641] highlighted the legal suitability of technical mechanisms for sanitizations (including redactions). However, in detail the analysis of legal texts yielded that in most a technically more precise description of the intended confidentiality protection was not found (see Analysis Result 16[642]). Not surprisingly, they provide the intended result of confidentiality using technology-neutral high-level descriptions in natural language[643]. In the US regulation this work found a pointer to a more technical document, the NIST Special Publication 800-88. However, the analysis of this yielded no technical input to answer the question as NIST SP 800-88 is focussed on the deletion of data from storage media including the physical destruction of the storage medium (see Sec. 9.3.4). Still, and this is different to integrity, the legal goal of the protection is — judging from the analysed definitions — unambiguously and unanimously agreed, as shown in

[637] In theory any other mechanism that achieves the same functionality as the MSS can be used to fulfil this requirement. For simplicity and increased readability this section will follow the focus of this thesis and talk about the mechanism as if it is an MSS. However, the requirement is formulated to be general.

[638] The privacy notion was described on a high-level in Sec. 3.9 and the existing MSS work was analysed in Sec. 9.1.

[639] Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. (see Sec. 9.4.1 on page 233).

[640] Analysis Result 21: The definition of the cryptographic property of standard privacy (see Sec. 9.4.2 on page 233).

[641] Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism (see Sec. 8.4.6 on page 211).

[642] Analysis Result 16: Legal texts acknowledge the importance of confidentiality protection of personal data and the high-level descriptions of the goal of confidentiality coincide (see Sec. 8.4.3 on page 208).

[643] For example "[...] rendering the stored data unrecognisable [...]" [BDSG]; see discussion in Sec. 8.2.5.

Analysis Result 16[644]. Regarding trade secret the protection goals also coincide in the analysed texts as Analysis Result 15[645] noted. Finally, this thesis found no clearly notable discrepancy between the technical confidentiality notions of indistinguishability and the technically more high-level descriptions, as summarised in Analysis Result 20[646].

In the following, this section will align cryptographic privacy definitions and legal confidentiality goals to formulate the requirement. The non-recoverable redaction or overwriting of data by sanitizations and thus the removal of information contained within can be seen as a mechanism for data minimisation. This is in line with the the aim of MSS. Thus, this thesis positions them as a tool for data minimisation, i.e. a privacy-enhancing technology (PET). The ENISA's report on the recommended cryptographic measures to secure personal data [187] states, in relation to the data minimisation principle (see Sec. 8.2.6 or Sec. 8.2.7) and informational self-determination: "In this definition, several advanced cryptographic schemes can play an important role, e.g. techniques to reduce the amount of personal data that is released to the strictly required minimum." [187]. The report from ENISA however does not list MSS as cryptographic techniques, the one page in the appendix of [187] describes algorithms to local extraction, attribute-based credentials, private information retrieval but makes no claim of these being the only mechanisms. In general, the authors of the report noted that reducing the amount of personal data helps "[...] reducing the need for "reaction", i.e. securing stored data." [187]. Following the GDPR the entity, "[...] which [...] determines the purposes and means of the processing of personal data [...]" [Art. 4 GDPR], "[...] shall implement appropriate technical and organizational measures to ensure and to be able to demonstrate that processing is performed in accordance with this Regulation."[647] [Art. 24 GDPR].

The standard privacy definition, which is the baseline privacy notion for this thesis, is compatible with this indistinguishability notion, as discussed in Sec. 9.1.5 and mentioned by *Brzuska et al.* in [64]. Thus, the confidentiality protection legally assumed to be offered by the sanitization or redaction is to achieve the erasure and thereby render the stored personal data unrecognisable[648]. Technically, the technique of sanitization or redaction has been explicitly named and described as well. Namely, ISO 27040 and ISO 27050 explicitly describe the goal of sanitization to either "render access [...] [to information] infeasible" [257] or remove the data containing the information "such that data recovery is not possible" [261].[649] Not unexpected, the more general standards and technical notions leave the fine tuning of the protection's strength to the application domain. Namely, both standards proclaim that the mechanisms must withstand an adversary at "a given level of effort" [257, 261]. Hence, to state that the strength of the adversary which the actual mechanism can withstand is not within these standards' scopes.

ENISA, also in the guidance on personal data [187], points to its periodical key size and algorithm report when saying: "When using any cryptographic mechanisms, in order to achieve a desired level of security, we recommend referring to the detailed information provided in [the ENISA report on Algorithms, key size and parameters][...]"[650] [187]. Note, the moment sophisticated attacks are known to a scheme or a the attack reduces the protection offered by some length of the key those schemes' usage is discourage in the ENISA reports[651]. All periodically issued recommendations on security strength of algorithms and the recommended key length, e.g., ENISA[651], Germany's *catalogue of algorithms*[652], and NIST SP 800-131A[653] do no longer encourage the use of an algorithm once an attack is known.

[644] Analysis Result 16: Legal texts acknowledge the importance of confidentiality protection of personal data and the high-level descriptions of the goal of confidentiality coincide (see Sec. 8.4.3 on page 208).

[645] Analysis Result 15: Definition of trade secret (in Sec. 8.4.2 on page 207).

[646] Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. (see Sec. 9.4.1 on page 233).

[647] The American English spelling from original work is retained for terms and quotes.

[648] In line with the notion of "erasure" from the German BDSG as analysed in Sec. 8.2.5 and the legal goal of confidentiality found for example in GDPR as summarised in Sec. 8.4.3 as Analysis Result 16.

[649] The notions are in line with the understanding of a valid sanitization used in this thesis, which was given in Definition 47 on page 49.

[650] The reference pointed to the 2013 version of the report, like [188] is the 2014 version.

[651] See Sec. 5.3.2.

[652] See Sec. 5.3.1.

[653] See Sec. 5.3.3.

This allows to formulate a requirement as follows. Note, the argumentation for an MSS offering a legally compliant strength becomes easier if the underlying cryptographic primitives embodied in MSS are already legally recognised and assumed to be strong primitives for the protection of personal data. This is the case for the indistinguishability property of encryption as algorithms which fulfil this property are registered as suitable in legal sources.

A MSS privacy property defined on the basis of the indistinguishability property for encryption allows to use related legal sources for the strength of the algorithms; Information-theoretic removal is the strongest level of protection. Under the GDPR the protection of the confidentiality of personal data through encryption does not remove the need to treat the encrypted cipher text still as personal data.[a] This thesis, however notes that confidentiality protection by encryption is a legally know and accepted protection mechanism[b]. The goal of confidentiality fulfilled by cryptographic encryption schemes is also defined on the notion of indistinguishability (see Sec. 9.2.4). For MSS the privacy notion is technically comparable and is also based on the indistinguishability (see Analysis Result 20[c]). As a result, this thesis proposes to use a privacy notion that is based on cryptographic indistinguishability and regard this as a legally suitable strength.

ENISA's report notes that "[o]ne can argue that practical cryptography is in fact the discipline that seeks to provide non-perfect but still good security while minimizing the size of the keys."[d] [187]. This note appoints existing legally accepted guidance on the length of keys and choice of algorithms for encryption as a reference when judging the suitability of the strength of MSS to withstand the recovery. Note, leakage through the semantics or context of the remaining message is not considered in MSS. Still, MSS need to reduce the amount of leakage that occurs through the adjusted signature valid on the modified message[e]. In CDSS the information flows from the data to the signature via the cryptographic hash-function. In CDSS, having information flow from the signed message's content into the hash-value is not a problem, as the message is not considered confidential. In MSS, however, it is concerned a leakage once the message's block has been subject to sanitization. Achieving perfect secrecy definitions, like the one of *Maurer* [331], then leads to legal data protection compliance of mechanisms. Such mechanisms would information-theoretically remove all traces of the data that got sanitized or redacted from the adjusted digital signature.

[a] Following Art. 32 GDPR the mechanisms of "pseudonymisation and encryption of personal data" [Art. 32, Par. 1, Item (a) GDPR] help "to ensure a level of security appropriate to the risk" [Art. 32, Par. 1 GDPR].

[b] For example, the GDPR identifies "encryption" as an "[...] appropriate technical [...] protection measures [...] that render[s] the personal data unintelligible to any person who is not authorized to access it [...]" [Art. 34, Par. 3, Item (a) GDPR].

[c] Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. (see Sec. 9.4.1 on page 233).

[d] The American English spelling from original work is retained for terms and quotes.

[e] This thesis uses the term adjusted signature to refer to the σ' from message-signature pair (m',σ') that is generated by a valid redaction or sanitization. The term modified message is used to refer to m' from message-signature pair (m',σ') that is generated by a valid redaction or sanitization.

10.1.3. Notes on achieving Requirement 9

This thesis offers in Sec. 17.1.3 the evaluation that cryptographic privacy with a strength at least as that defined by *Brzuska et al.* [64] is sufficient to fulfil Requirement 9. The definition were given in a harmonised notation in this thesis in Sec. 11.6.3 for SSS and in Sec. 11.13.3 for RSS.

10.2. Requirement 10 (Mechanism MAY prohibit the identification of additional context information further to Requirement 9 without violating Requirement 4)

This final requirement is special, as it captures explicitly an implication due to the combination of integrity protection requirements and confidentiality requirements from this thesis' research question. As a 'cross-implication' this is neither a requirement to deliver a compliant level of integrity protection nor of confidentiality for increased privacy it is optional. This is indicated by the term 'MAY'. It captures as technically precise as possible that the mechanism could provide a stronger notion than standard privacy as captured in Requirement 9, as long as this does not violate the diametrical positioned Requirement 4[654], because it is central to an increased probative value.

To follow the requirement's suggestion that the mechanism SHOULD achieve non-interactive public accountability of the Signer to allow the legally preferred public form of accountability (PUB) leads to the exclusion of transparency as a stronger notion compared to standard privacy. However, schemes that offer additional privacy guarantees while still offering non-interactive public accountability (PUB) are considered in this thesis.[655]

10.3. Linkage between technical and legal requirements for confidentiality and analysis results

Following the methodology, the analysis results have influenced the technical requirements for confidentiality protection. The analysis resulted in the legal definitions for personal data [656] and trade secrets[657] used in this thesis. They further yielded the thesis' technical definition of the cryptographic property termed standard privacy[658] based on the cryptographic notion of indistinguishability[659] known from the realm of confidentiality of encryption. Analysis Result 16[660]Analysis Result 17[661], and Analysis Result 18[662] additionally motivate the use of MSS in the legal context. The final legal Analysis Result 19[663] establishes that the functionality of redaction and sanitization inherent to MSS are actually a legally accepted tool. Taking this into account, Requirement 9 is worded such that it allows a technical enforcement with suitable RSS and SSS mechanisms. Overall, all analysis results influenced the baseline confidentiality requirement (Requirement 9) presented as a result of Part II. As Requirement 10 is concerned with cross-implications between integrity requirements vital to probative value and stronger privacy it was also influenced in manyfold ways by the analysis results in both areas of interest.

[654] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[655] Note, as discussed in Sec. 7.4 a signature scheme offering the cryptographic property of transparency would mean that authorized subsequent modifications stay hidden from the Verifier unless the Signer interactively agrees to make them detectable. As those schemes — by their properties — are not nearly as technically aligned to legally accepted schemes (CDSS) they might or might not offer the same increased probative value. In Chapter 17 the analysis argues on technical equivalence and thus the public form of accountability (PUB) is preferred. This is, however, not withstanding that schemes with ACA – INT could in theory be argued to fulfil all legal requirements towards integrity.

[656] Analysis Result 14: Definition of personal data (see Sec. 8.4.1).

[657] Analysis Result 15: Definition of trade secret (see Sec. 8.4.2).

[658] Analysis Result 21: The definition of the cryptographic property of standard privacy (see Sec. 9.4.2).

[659] Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. (see Sec. 9.4.1).

[660] Analysis Result 16: Legal texts acknowledge the importance of confidentiality protection of personal data and the high-level descriptions of the goal of confidentiality coincide (see Sec. 8.4.3 on page 208).

[661] Analysis Result 17: Legal texts demand the confidentiality protection of trade secrets if not outweighed by common interests (in Sec. 8.4.4 on page 209).

[662] Analysis Result 18: The importance of or the right to confidentiality protection for data might vary over time (see Sec. 8.4.5 on page 210).

[663] Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism (see Sec. 8.4.6 on page 211).

PART III

Proposed constructions of private redactable and sanitizable signatures with an increased probative value

Butler Lampson, a well-known and distinguished American computer scientist, says, "Anybody who asserts that a problem is readily solved by encryption, understands neither encryption nor the problem."

— R. M. Needham
Speech at the Marshall Symposium, 1998 [357]

Contents

Chapter 11 State of the Art: Harmonised notation of existing security properties for SSS and RSS

Chapter 12 Shortcomings of state-of-the-art SSS and RSS with respect to the requirements

Chapter 13 Proposed private SSS with an increased probative value

Chapter 14 Proposed private RSS with an increased probative value

Chapter 15 Separating Secure RSS from Secure SSS

11 — State of the Art: Harmonised notation of existing security properties for SSS and RSS

Overview of Chapter 11

The greater context of Malleable Signature Schemes has already been given in Sec. 3.4. Notational preliminaries, applicable to both analysed concepts RSS and SSS, can be found in Sec. 3.5. This notation is used for this chapter's harmonised presentation of the existing security properties of RSS and SSS. This takes into account the state of the art and presents them in a coherent notation and security model.

In more detail this chapter states for SSS and RSS:

- **State of the art**: Brief background on the existing works that this thesis has considered. The works considered especially important or of extended interest have already been mentioned or will get mentioned in more detail when their contributions are discussed in more depth at different locations throughout this thesis. The existing work in the field of SSS is laid out in Sec. 11.1. For RSS the existing body of work is presented in Sec. 11.8.
- **Harmonised generic algorithmic description**: A presentation of the algorithms in a harmonised way; see Sec. 11.3 for SSS and Sec. 11.10 for RSS. This includes the formalisation of sequential execution and correctness properties.
- **Presentation of existing security properties in a harmonised notation**: The existing security properties from the state of the art are presented in a harmonised notation; each property's description incorporates the latest findings regarding known attacks or refinements and thus consolidates the existing works in the area. See Sec. 11.6 for SSS and Sec. 11.13 for RSS. This includes a presentation of the relations between the state-of-the-art properties.

The state of the art discussion is followed by a list of shortcomings in Chapter 12. Eight new schemes with new properties that address the shortcomings are proposed in Chapter 13 for SSS and in Chapter 14 for RSS. Finally, Part III concludes with Chapter 15 formally separating RSS and SSS as two distinct forms of MSS.

11.1. SSS: State of the art

A history of the terminology given in the literature was already presented in 3.4.3. In this section the works known to the author of this thesis in the field of SSS are presented[664]. They are sorted by year of publication. Works considered especially important or further links between them that are of extended interest have already been mentioned or will get mentioned in more detail when their contributions are discussed in more depth at different locations throughout this thesis. The state of the art in RSS is presented in Sec. 11.8. RSS get cryptographically separated from SSS in Chapter 15.

There is related work in many areas around digital signatures. For example, in 2002 *Hevia and Micciancio* [229] constructed a graph-based one-time signature. Their work explores that those signatures would allow arbitrary parties to perform a limited set of operations to alter the signed data in logical ways. They state that this is "[...] a new paradigm for the construction of algebraic signature schemes, which are new useful primitives for applications where controlled "forgeability" of signatures is needed, as in credential systems." [229]. Their work, like others is classified as being in the direction of allowing mathematical functions as subsequent operations, this is seen as a different class of signature schemes. As there are differences as well as overlaps between other schemes and the RSS and SSS discussed in this thesis the classification is not sharp. However, the following section discusses schemes that are concerned with MSS where the modifications that can be authorized lend themselves to documents or blocks thereof. See [137, 475] for an overview of so-called 'functional signature schemes' or 'homomorphic signature schemes' that are geared towards arithmetic subsequent modifications.

Note, own or joint work involving the author of this thesis is not listed in this section; an overview of joint works is in Appendix A and for a brief discussion how that work influenced the state of the art see Sec. 16.4.

In 1998, *Krawczyk and Rabin* [291] published their work, dated 1997 on the cover, on so-called chameleon hashes. It was also published two years later as [292]. For a chameleon hash function a trapdoor exists which allows to efficiently find collisions, without knowledge of the trapdoor "[...] the function is collision resistant in the usual sense, namely, it is infeasible to find two inputs which are mapped to the same output." [291, 292]. Their application makes the Signer choose a recipient and allow this recipient to be the Sanitizer by incorporating a trapdoor into the signature that would allow the recipient to find collisions to the message, but not the Signer. Thus a resulting valid signature corroborates to the verifying recipient that it has been generated by the Signer. They used the ability that a party could use the trapdoor to create a hash collision to build "undeniable signatures". Their scheme offers "non-transferability" [291, 292]. "Given a signature [...] [and a message] generated by the signer S for a recipient R, the recipient cannot convince a third party of its validity. That is due to the fact that for every possible message m', R can compute [...] a proper [valid] signature." [291, 292]. The original concept of chameleon hashes for signatures by *Krawczyk and Rabin* [291], is considered the basic building block that was later extended by *Ateniese, Chou, de Medeiros, and Tsudik* [12] to be applied to blocks of messages for which *Ateniese et al.* coined the term 'sanitizable signatures'. Note, the undeniable signature scheme described in [291, 292] does not offer the public detection of subsequent changes, i.e. the scheme offers ACA – UCA – PUB. Only unauthorized modifications are publicly detected, because a party not having the secret signing key or the secret trapdoor for sanitizing cannot compute a valid signature and thus the Verifier will notice the modification as signature verification will fail on the forged message. As indeed a Sanitizer is authorized to modify the message in any way another entity can not be convinced of the Signer being the origin unless running an interactive protocol. If in this interactive protocol the Signer is forced to always correctly generate a proof to repudiate being the origin of a genuinely sanitized message or being accountable for the original message, then a third party running the interactive protocol as a Verifier can indeed become convinced of the accountable party. However, this is only possible in interaction with the Signer, which can be beneficial as motivated in [291, 292]. This is noted as ACA – ≥1CD – INT integrity protection. For a discussion on the enforcement of correct execution and non-repudiation see Sec. 6.4.1. Note, with ACA – ≥1CD – INT and a correctly-executed and non-repudiable interactive dispute resolution protocol the signature becomes transferable, while still retaining the need for interaction for the first generation of this proof. This, and other techniques, are discussed by *Krawczyk and Rabin* [291] when discussing the conversion of "[...] an undeniable signature [...] into a regular publicly verifiable (non-repudiable) signature [...]" [291, 292] following [60]. In the applications discussed in this thesis the entity in the role of the Verifier is not the

[664] This list contains work roughly till the third quarter of 2017.

same as the Sanitizer; and the legal analysis has shown, that rather than an interactive protocol, a public form of 3^{rd}-party verifiability is beneficial for the Signer to increase probative value towards the Verifier. Hence, the thesis constructions' goal is to establish accountability solely from public information[665].

In 2000, *Rivest* [407] presented the idea that signature schemes could allow special pre-defined forgeries in a talk[666]. One example given in [407] for pre-defined allowed subsequent manipulation of signed data was the multiplicative behaviour of RSA. *Rivest* presented the adapted definition of the security in the talk and introduced two schemes: One scheme on transitive signatures [339] got published in 2002. The second remained unpublished and is a signature scheme for route aggregation [118]. The latter is closer to a redactable scheme. *Rivest* did not mention the impact the ability for limited subsequent manipulation of signed data has on the notion of integrity [407].

Krawczyk and Rabin [293] filed a patent in 1998 which was published in 2000 [293]. The patent contains the method published first on ePrint in [291] which later appeared in NDSS 2000 [292].

In 2002, the idea of letting parties subsequently perform a limited and defined set of operations on signed data without invalidating the signature and without access to the secret signature generation key got formalised in a number of papers.

Micali and Rivest [339] presented work on transitive signatures for graphs that followed the ideas presented by *Rivest* in 2000: "Informally, this is a way to digitally sign vertices and edges of a dynamically growing, transitively closed, graph G so as to guarantee the following properties: Given the signatures of edges (u, v) and (v, w), anyone can easily derive the digital signature of the edge (u, w)." [339]. This type of signatures is related as it fits the general idea that an MSS is able to generate valid signatures for defined modified data from existing validly signed data without access to the signature generation key.

Chari, Rabin, and Rivest [118] further presented work on a signature working on a binary tree structure that would prohibit an adversary to produce valid signatures on a node "without the signatures on [both] its children" [118]. This thesis classifies this as more in the direction of RSS for two reasons: First, the construction leans more into the direction of removing the need to retain and verify signatures on all child nodes, even though the use in the application of routing allows to combine information. Second, the authors compare themselves and cite influential work on redactable signatures from *Johnson, Molnar, Song, and Wagner* [273] — also published 2002 — as related work.

In 2003, *Miyazaki, Susaki, Iwamura, Matsumoto, Sasaki, and Yoshiura*' work [346] is titled "Digital documents sanitizing problem" [346]. While it is in Japanese this thesis assumes that it describes in fact an RSS; this is deducted from the authors' later work [347] in 2005. This later work also has the word 'sanitizing' in the title but only to describe the more generic outcome of the action of redaction.

In 2005, *Ateniese, Chou, de Medeiros, and Tsudik* [12] titled their work using the term 'sanitizable signature scheme' (SSS). Their SSS allowed arbitrary modification of pre-determined blocks of signed messages. This was cryptographically built on top of chameleon hashes proposed by *Krawczyk and Rabin* [292]. This was the change over the existing works, e.g., from 2002. Previous works only allowed a limited set of operations. *Ateniese et al.* also noted that the capability of a subsequent modification is important for many applications; e.g., authenticated multicast, database outsourcing, and secure routing. *Ateniese et al.* [12] introduced the five security properties, namely privacy, immutability, accountability, transparency and unforgeability. However, they did not give formal definitions for these properties. Most properties[667] have been formalised and extended by *Brzuska et al.* [64] in 2009. *Ateniese et al.*'s sanitizable signature scheme [12] is based on chameleon hash-functions [292] that were applied per each admissible block. Note that their construction allows for mix-and-match attacks[668]. To solve this,

[665] Information like message, signature, auxiliary system parameters or public keys are considered publicly available to a Verifier.

[666] This cite was picked up from the work of *Ateniese, Chou, de Medeiros, and Tsudik* [12].

[667] Some properties of [12] were not formalised in [64], e.g., invisible sanitizable signature scheme was only recently formalised by *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* at PKC 2017 as joint work [107] (see Appendix A publication n° 55).

[668] For a mix-and-match attack the adversary creates a forgery by building a message from blocks which have all appeared in genuinely signed messages, but have never appeared together in a signed document (description adapted from [455]).

the later schemes based on the same general idea required to find a collision on each admissible block, not only those modified, like the scheme presented by *Brzuska et al.* [64]. Note further that *Ateniese et al.* mention the term ‘integrity’ and state that “[s]anitizable signatures can be used to ensure the integrity, authenticity, and anonymity of PHI [Personal Health Information] [...]” [12]. However, they do not discuss the impact of the permission of authorized subsequent changes on the protection goal of integrity in more detail.

Izu et al. [264] is titled “A Partially Sanitizable Signature Scheme” [264] but it is only allowing for the removal of blocks. Hence, despite of the work’s title, this thesis deems this scheme to be an RSS. The same holds true for *Miyazaki et al.* [347] which also has the word ‘sanitizing’ in the title to describe the more generic outcome of the action of redaction. Their work titled “Digitally Signed Document Sanitizing Scheme with Disclosure Condition Control” [347] describes in fact an RSS.

Bellare and Neven [36] continued the work on the topic of transitive signatures. Note that transitive signatures are only related to RSS and SSS. The transitive signatures from [36] work on graphs and allow to derive signatures on the transitive closure of a path in a graph from individual signatures on all edges within the path. The authors mention the work by *Micali and Rivest* [339] from 2002 in their motivation, so they share some of the motivation. However, this thesis sees them as different mechanisms, as their underlying idea is to combine signed data. This is different from the explicit idea to alter data inherent in SSS or in RSS.

In 2006, the scheme of *Klonowski and Lauks* [285] allows for arbitrary modification of pre-determined submessages and limits “[...] the censor’s abilities to modify mutable parts of a signed message to a predetermined set of strings.” [285]. They further note that “[o]ne of the most important problems with regular sanitizable signatures [...] is that the censor can exchange [a] particular block in [an] arbitrary way. In practice the signer often needs to give only limited list of possible blocks that can be inserted in particular position in the text. ” [285]. The authors propose a solution on bloom filters, citing [54], and one on cryptographic accumulators, citing [43]. The latter is also a common cryptographic building block in the area of RSS[669]. Their idea got picked up and formalised in 2010 by *Canard and Jambert* [109].

In 2008, *Yuen, Susilo, Liu, and Mu* [508] made an effort to combine the models for RSS and SSS which existed at that time and provide a first idea on how those relate. They state that a scheme offering SSS functionality where blocks can be sanitized into arbitrary messages can be converted into a scheme offering RSS functionality by using a special character $\emptyset$ [508]. The authors neglect to discuss the impact this would have on the property of privacy, even though they discuss transparency. As this thesis has discussed in Sec. 9.1.6 their privacy and transparency notion are not fully compatible with the notion given later by *Brzuska et al.* [64] and chosen for this thesis. Further this thesis shows in Chapter 15 that RSS and SSS differ in the privacy property too much to be black-box transformed as easy as the reader of [508] might think. Further, the scheme presented in [508] was shown to not achieve a sophisticated level of transparency following the work from 2009 by *Agrawal et al.* [4].

Canard, Laguillaumie, and Milhau [111] presented a scheme that allows for arbitrary modifications of pre-determined blocks of a signed message. *Canard et al.* in this work [111] and in their later work [112] did not require the Signer to define all admissible blocks of the signature during signature generation. Instead they took another approach, “[...] the power of sanitization is given to possibly several entities, for a given message/signature by using a trapdoor computed by the signer at any time.” [111]. This includes that there are several Sanitizers instead of one and that ADM is dynamically managed by the Signer over time. Namely, the policy might increase the ability of Sanitizers, instead of keeping it or reducing the Signer defined policy which is discussed under the notion of ‘consecutive redaction control’ (see Sec. 14.5.1.1). This thesis analysed the legal impact of MSS in the single Sanitizer setting. An extension to multiple Sanitizer let alone a dynamically increasing policy is left as future work (see Sec. 19.3).

[669] This was already noted by *Klonowski and Lauks* by stating “Let us note that extended accumulator idea was previously used in [455] in another context to built redactable signature scheme” (cited reference was adjusted) [285].

Haber, Hatano, Honda, Horne, Miyazaki, Sander, Tezoku, and Yao's work [222] is leaning more towards an RSS than an SSS, however as it allows for a limitation of the values that can be inserted by Sanitizers it is mentioned here. Namely, in their scheme "[...] a zip code 32578 could be generalized to 3257*, and further to 325**, etc; a date could be generalized from June 10, 1968 to June 1968, to 1968, then to 1960-1969. Generalization and suppression can be modeled as trees, where a suppressed attribute corresponds to a NULL element at the root of the tree."[670] [222]. Note, they also mention the property of consecutive redaction control known to RSS from [347]. As their scheme is considered closer to an RSS it gets mentioned again in Sec. 11.8.

Izu, Kunihiro, Ohta, Sano, and Takenaka [265] presented combinations of the approaches from sanitizable and redactable signatures. It is discussed in the state of the art of RSS. This thesis provides a closer look on the differences between RSS and SSS and formally separates them (see Chapter 15).

Chang and Xu [116] showed in their work on remote integrity checking for dishonest storage servers that an RSS can provide useful functionality as one of their building blocks. Their construction incorporates an RSS which they base on the scheme presented in *Johnson et al.* [273].

In 2009, *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk* [64] closed the gap that no overall formalisation of the SSS properties was given in 2005 by *Ateniese et al.* [12]. Note, they also slightly extended the original properties of *Ateniese et al.*[671]. In line with the scheme of *Ateniese et al.* — on which they based their work — *Brzuska et al.* [64] allowed arbitrary modifications of pre-determined blocks of a signed message. The formal definitions of unforgeability and accountability presented are not met any more by *Ateniese et al.*'s original construction. [64] To prevent the mix-and-match attacks on the original scheme *Brzuska et al.* introduced the need to find a collision on each admissible block, not only for those modified. This makes the sanitizing process' overhead linear in the number of admissible blocks. Note that *Ateniese et al.* define an SSS as secure if it comes with the five security properties of unforgeability, immutability, privacy, accountability and transparency. While *Brzuska et al.* [64] state that a meaningful SSS needs not necessarily satisfy transparency [64].

Brzuska, Fischlin, Lehmann, and Schröder [65] also showed in 2009 "that by sacrificing the transparency property [...] [one] can obtain a sanitizable signature scheme that is still provably secure concerning the other aforementioned properties [unforgeability, immutability, privacy and accountability] but significantly more efficient." [65]. They proposed an SSS with a changed underlying paradigm: [65] proposes two interleaving signatures instead of an operation on each admissible block. Previous schemes had to find a collision on each admissible block, not only those modified [64]. Thereby, the algorithms involving messages become constant and are no longer linear in the number of admissible and fixed blocks.

Agrawal, Kumar, Shareef, and Rangan [4] provides increased privacy for SSS by providing "two protocols for strong transparency in the standard model using bilinear pairing" [4]. The authors' work states that it "uses techniques similar to the sanitizable signature scheme discussed in [508] and provides additional properties. The scheme in [508] does not provide transparency." (the cited reference was adjusted) [4]. Thus they improved the body of work that allows for transparency by rectifying a shortcoming from the 2008 work of *Yuen et al.* in [508]. The authors introduced two levels of transparency which are not fully compatible with other formalisations as discussed in Sec. 9.1.6.

In 2010, *Canard and Jambert* [109] proposed a scheme that allows for arbitrary modifications of predetermined blocks of a signed message with the extension to limit Sanitizers to certain values. Hereby, *Canard and Jambert* provides a formal model for the proposed idea from *Klonowski and Lauks* [285] from 2006. The following additional limits exist according to [109, 285] for the Sanitizer: (1) being limited to a fixed set of new values per blocks, (2) being limited to choose the same values for specifically linked blocks, (3) being limited in the total number of modified blocks per a single sanitization of the message, and (4) being limited in the number of parallel versions created by valid authorized modifications. In 2015 the work from *Derler and Slamanig* [139] shows that the formalisation for privacy given in [109] has not been precisely enough defined.

[670] The American English spelling from original work is retained for terms and quotes.

[671] This analysis was done during this thesis and the results got published as [428] (see Appendix A publication n° 27).

Brzuska, Fischlin, Lehmann, and Schroeder [67] introduced the concept of unlinkability, a privacy notion which prohibits a third party from linking two messages [67]. This work instantiated the SSS security property of unlinkability defined in the harmonised notation in Sec. 11.6.5. The proposed SSS offers unlinkability and it follows in its construction the underlying paradigm from [65]. In [67] from 2010 *Brzuska et al.* based their scheme on two signatures: One to sign the fixed parts of m, i.e. $m[\mathsf{fix}]$, sometimes also denoted as the "inner signature", and another one to sign the admissible parts, i.e. $m[\mathsf{ADM}]$, together with the fixed parts, often called the "outer signature". The inner signature is produced by the Signer of the signature scheme, while the outer signature can be produced by either one, the Signer or the Sanitizer. Thereby, this scheme also offers algorithms for sign, sanitize and verify that are constant in the number of admissible blocks of the message. Using different signature types as inner and outer signature yields different properties of the sanitizable signature scheme. For instance, *Brzuska et al.* used a group signature for the outer signature in [67]. The anonymity of the group signature makes signatures of the signer and the sanitizer indistinguishable, and the non-frameability/traceability property of the group signature scheme assures an interactive form of accountability. However, to achieve the property of unlinkability combined with transparency, this requires the costly utilisation of group signatures [67].

Yum, Seo, and Lee [510] offer an analysis what is minimally needed, i.e., minimal computational complexity assumption, to build a sanitizable signature mechanism: "[...] one-way functions imply trapdoor sanitizable signatures." [510]. The authors present a generic construction from ordinary signature schemes that they instantiate using " [...] known identity-based chameleon hash functions [...] based on the RSA signature or the bilinear maps [...]", e.g., [11] is cited by *Yum et al.* [510].

In 2011, *Yum and Lee* [509] reconsidered SSS and optimised their construction to not apply the cryptographic operation to each blocks. Their scheme offers a computational complexity that is constant and not linear in the number of blocks. The resulting construction in [509] deploys labels and achieves also a constant signature size. This general idea got reconsidered when developing the construction for strengthened unlinkability realised by the $unlinkable\mathcal{SSS}$ scheme even without the need for labels (see Sec. 13.9.2).

Gong, Qian, and Zhou [216] showed problems in the construction of *Brzuska et al.* [64] and *Canard and Jambert* [109]. They present that "[i]t is easy to overcome this flaw [...]" [216]. Especially, they "[p]oint out the weakness of the existing security model" by *Brzuska et al.* [64]. The attack presented would allow an adversary to forge the policy ("rights forge attack" [216]) and as the the security model of [64] does "'[...] not take the description ADM [...] into account and thus [...] [it] fails to capture the corresponding potential attacks [...]" [216]. As a fix it is required for all SSS that the description of the admissible parts ADM is recoverable from any valid message-signature pair (m, σ). And ADM then must be taken into account in the game-based definitions of the following security properties: unforgeability, immutability and accountability. *Gong et al.* [216] present a construction that is transparent and secure in [64]'s rectified model, so it achieves Signer- and Sanitizer accountability but not in its public form, i.e., ACA – ≥1CD – INT.

In 2012, *Ahn, Boneh, Camenisch, Hohenberger, abhi shelat, and Waters* [6] introduced a stronger notion of statistical unlinkability, but they did so for *redactable* signature schemes (RSS). This scheme was already released as [5] and influenced the work in the following years. *Attrapadung, Libert, and Peters* [15] introduced also a stronger privacy notion for RSS. As both works are in the domain of RSS the details are discussed in Sec. 11.8. They are listed here to note that neither of the aforementioned strengthened notions (statistical transparency and statistical unlinkability) have been considered in the context of sanitizable signature schemes. This gap is addressed in this thesis by the construction of $unlinkable\mathcal{SSS}$ offering the property of strengthened unlinkability for SSS (see Sec. 13.3).[672]

Canard, Jambert, and Lescuyer [112] in 2012 extended the formalisations from *Brzuska et al.* [64, 67] into a framework for multi-Sanitizer environments. Previous, the works did only "[...] consider one single signer allowing one single sanitizer to sanitize a given message-signature pair" [112]. Following *Canard et al.* "[a] signer can choose several designated sanitizers for a given message and each of them is able to modify the resulting message-signature pair." [112]. They present a multi-user framework which works for n Signers and m Sanitizers and offers the "traditional procedures [...] and security pro-

[672] The scheme and the property was published in 2013 as joint work with *K. Samelin* and *C. Brzuska* (see Appendix A publication n° 17).

perties: immutability, signer and sanitizer accountabilities and transparency [...]" [112]. They adjust the immutability, accountability and transparency notation to cater for the added parties. Regarding the latter, note that the transparency property by *Brzuska et al.* [64] for standard single-Signer single-Sanitizer SSS translates into *Canard et al.*'s notion of "[...] full transparency where this can only be done by the true signer, the true sanitizer and a designated authority." [112]. This thesis works within the framework for a single Signer and more specifically only for a single Sanitizer. In general, the thesis left it as future work to extend constructions and properties to the multi-user setting from [112] (see Sec. 19.3). Some ideas on how certain schemes can be extended towards multiple Sanitizers have been explained in joint research papers [395] (see Appendix A publication n^{o} 15) or for the scheme $unlinkable\mathcal{SSS}$ [69] (see Appendix A publication n^{o} 17).

In 2013, *Hanser and Slamanig* [224] introduced a concept they termed 'blank digital signatures' as follows: "The basic idea behind [their] scheme is that an originator can define and sign a message template, describing fixed parts of a message as well as multiple choices for exchangeable parts of a message. One may think of a form with blank fields, where for such fields the originator specifies all the allowed strings to choose from. Then, a proxy is given the power to sign an instantiation of the template signed by the originator by using some secret information. By an instantiation, the proxy commits to one allowed choice per blank field in the template. The resulting message signature can be publicly verified under the originator's and the proxy's signature verification keys. Thereby [due to the MSS like privacy property] no verifying party except the originator and the proxy learn [sic] anything about the 'unused' choices from the message template given a message signature. Consequently, the template is hidden from verifiers." [224]. Apart from [224] also the work of *Derler et al.* [140] from 2014 gives an instantiation. This thesis had discussed the legal implications of blanket statements in Sec. 4.5.5. How SSS providing a high probative value can — also following [224] — indeed be used to technically construct such blanket statements is described in Sec. 18.7.

Lai, Ding, and Wu [304] in their paper on accountable trapdoor sanitizable signatures couple the sanitization step to an interaction. In the flow of an SSS used in this thesis, the Sanitizer is fixed during the signature generation but can compute valid sanitizations on signed messages at any time. Especially, in this thesis the sanitization was assumed to not require any interaction with the Signer. *Lai et al.* [304] remove this decoupled process: "This signature alone does not allow the candidate sanitizer to produce a new signature on a sanitized message. To generate a new signature on a sanitized message, the candidate sanitizer needs to obtain a trapdoor from the original signer."[304]. This removes some of the flexibility, e.g. useful in the case where also a re-signing cannot be done as the original Signer should not or can no longer be contacted. Of course *Lai et al.* [304]'s scheme hands additional control to the Signer. What might be needed might depend on the specific target application. They provide constructions that still provide Signer and Sanitizer-accountability, while being transparent to the Verifier unless executing an interactive dispute resolution protocol. In line with this thesis and other works they state: "A Judge algorithm then uses the proof π to decide if a valid message-signature pair was created by the signer or the sanitizer (the lack of such a proof is interpreted as a signer origin)." [304]. Especially, the latter mentioned 'room for interpretation' requires a correctly-executed and non-repudiable communication including the Verifier's request for π and the Signer's response as discussed in Subsections 6.4.1 and 6.4.2.

Canard and Lescuyer [110] use the concept and security provided by sanitizable signature schemes to obtain "[...] certified credentials from organizations and use them later without being traced [...]"[673] [110]. "A user gets from the organization a [sanitizably] signed document certifying personal data [...] and plays the role of the sanitizer. When showing his credential, [t]he [user] uses sanitization techniques to hide the information [t]he [user] does not want to reveal [...] and shows the resulting document, which is still seen as a document certified by the organization."[673] [110]. The authors correctly note that the two primitives SSS and anonymous credentials (AC) have different functionality. In order to build AC the authors want the SSS to behave on one hand like an RSS with dedicated redactors and they state that they "[...] want the true sanitizer to be able to hide admissible parts of the message. However, we do not want him to modify as he wants these admissible parts [...]" [110]. On the other hand they want traceability, i.e. "[...] the necessity for an algorithm to recover the actual designated sanitizer of a given message-signature pair." [110]. This requires to identify the Sanitizer, which can be done in SSS due to the keyed operation. *Canard and Lescuyer* name this algorithm "FindSan" [110] and show that with

[673] The American English spelling from original work is retained for terms and quotes.

Sanitizer-accountability such an algorithm can be built. This thesis notes that this can be accomplished with ease if the SSS would fulfil a public form of accountability, i.e. non-interactive public accountability. Additionally the authors note that "[...] the anonymity level of the credential system depends on the unlinkability level of the underlying signature scheme." [110].

In 2014, *Chase et al.* [119] generally captured "[...] transformations (such as redactable [...], sanitizable [...], quotable [...], and transitive signatures" [119]. *Chase et al.* define a malleable signature as follows:

"A signature scheme is malleable — alternatively, homomorphic — if, given a signature σ on a message m, one can efficiently derive a signature σ' on a message $m' = T(m)$ for an allowed transformation T." [119] This is in line with the way this thesis has defined the notion of subsequent authorized modifications and in line with this thesis understanding of the actions of redact and sanitize.

Backes, Dagdelen, Fischlin, Gajek, Meiser, and Schröder [19] offer a generalisation. Instead of an SSS with the functionality to edit blocks, they describe what they termed 'operational signatures schemes'. This general notion then captures more broadly several signature schemes that allow to do subsequent operations (called predicates) including sanitizable and redactable signature schemes. They note that "[...] the concrete privacy requirement, if needed at all, for signatures and MACs varies significantly with the application [...]" [19]. The privacy notion they have chosen for the operational signatures is in line with the one favoured in this thesis, it is also based on the indistinguishability idea, but got generalised for predicates. The authors state that their privacy notion follows the meaning of those already known, i.e. the adversary "[...] cannot distinguish deduced signatures from fresh signatures [...]" [19]. They further explicitly state that it embraces the privacy notion of RSS given by *Brzuska et al.* [66].

In 2015, *Derler and Slamanig* [139] provided a more detailed analysis of the extension of sanitizable signatures which allowed to limit Sanitizers that got introduced in 2006 by *Klonowski and Lauks* [285] and got formalised by *Canard and Jambert* [109] in 2010. Due to its impact on privacy the authors concentrate on the restriction of the Sanitizer being limited to a fixed set of new values per block. This extension is also denoted as "LimitSet"[109, 139]. The authors motivate that for this extension the privacy notation for the SSS needs to be adjusted as a private set limited SSS shall not allow the Verifier to know the elements of the set of values. They note that the adjustment by *Canard and Jambert* [109] did not correctly cover this as "[...] the privacy guarantees of their [referring to *Canard and Jambert* [109]] model do not capture privacy in the sense of the original definition of sanitizable signatures. That is, if a scheme is private in this model it is not guaranteed that the sets of allowed modifications remain concealed." [139]. The authors noted that the strengthened notion of strong unlinkability as presented in Sec. 13.3 of this thesis is helpful as '[...] a stronger notion of privacy, i.e., (strong) unlinkability (defined by [*Brzuska et al.* [69]] at EuroPKI'13) [...] fixes this problem [...]" [139]. They quote [69], which is joint work with *C. Brzuska* and *K. Samelin* presented at EuroPKI 2013 (see Appendix A publication n$^{\mathrm{o}}$ 17) and contains the results from this thesis on strong unlinkability. Further *Derler and Slamanig* note that currently "[...] no efficient unlinkable scheme supporting the aforementioned extensions exists and it seems to be hard to construct such schemes. As a remedy, in this paper, we propose a notion stronger than privacy, but weaker than unlinkability, which captures privacy in the original sense." [139]. They point at the work of *Canard and Lescuyer* [110] from 2013 for a "rather inefficient" [139] instantiation that unfortunately "[...] is only proven secure in a customized model which does not consider all security requirements [...]"[674] [139]. Finally, *Derler and Slamanig* note that the often cited "implication of privacy by transparency" [139] from *Brzuska et al.* [67] "only holds in the proof-restricted case" [139].

Derler, Hanser, and Slamanig [142] continued to optimise and implement the constructions for 'blank signatures' as introduced in [224] in 2013.

Fehr and Fischlin [191] worked on sanitizing encrypted authentic data. Through this the Sanitizer is no longer required to know the original message-signature pair in clear. The authors' constructions achieve security properties with respect to privacy and accountability which are in-line with the privacy requirements for standard privacy, as they rely on the underlying SSS which they state could be "[...] to start with a perfectly private sanitizable scheme, such as in [*Brzuska et al.* [67]]."[191]. For accountability *Fehr and Fischlin* [191] explicitly cite published results of this thesis and mention that they target and achieve non-interactive public accountability "[...] which is achievable, as long as the underlying sanitizable signature scheme is publicly accountable, too." [191].

[674] The American English spelling from original work is retained for terms and quotes.

Gajek, Seedorf, and Dagdelen [201] applied for a patent for a 'method and system for modifying an authenticated and/or encrypted message' in 2013 which got granted in 2015. The method described combines a chameleon hash-function based SSS like — and citing — the one by *Ateniese, Chou, de Medeiros, and Tsudik* [12] with a per blocks encryption to keep the confidentiality of all blocks of the message. This allows the application of an SSS on an encrypted message as long as it contains individually decomposable and decrypt-able blocks. No further new properties are covered by the patent.

In 2016, *Fleischhacker, Krupp, Malavolta, Schneider, Schröder, and Simkin* [197] published their work on efficient unlinkable SSS that was presented in 2015 [196]. They built a more efficient scheme that is not based on group signatures as the original unlinkable construction from *Brzuska et al.* [67]. This work took note and cites several of the results that were timely published[675] during the work on this thesis. *Fleischhacker et al.* focus on "[...] on schemes that have (inter-active) accountability and transparency." [197]. This falls into the accountability notion INT.

Lai, Zhang, Chow, and Schröder [305] identified that the existing works on unlinkable SSS either rely on the costly primitives, like the group signatures of *Brzuska et al.* [67] or are only secure in the random oracle model (ROM), like the construction by *Fleischhacker, Krupp, Malavolta, Schneider, Schröder, and Simkin* [197]. The authors also state that the additional strengthening proposed in this thesis "to also conceal the sets of allowed modifications [69][676][...] appears to be difficult to construct [...] efficiently." [305]. The property is described in Sec. 13.3 in detail.[677] *Lai et al.* [305] cite — among others — published results from this thesis on non-interactive public accountability[678] when they state that their work has a "[...] focus on schemes that have (interactive) accountability and transparency." [305]. This means their scheme achieves unlinkability on top of ACA – $\geq$1CD – INT integrity.

In 2017, *Wang, Pang, and Deng* [498] allowed to model workflows, building on the capabilities of leaving a blocks of a message underspecified that is inherent to SSS. In line with the idea to build templates using SSSs for 'blank signatures' as proposed by *Hanser and Slamanig* [224] in 2013, the authors present a "cascade-instantiable blank signature (CBS)" [498] that caters to an "application scenarios involving a sequence of proxies. Here, each proxy in a delegation chain creates from her direct predecessor's template/instantiation a new instantiation that narrows the successors' choices for the exchangeable fields [...]" [498]. This application of flow control is in the direction of the application scenarios discussed in Sec. 18.5 and would benefit from knowing the legal probative value of the proxy generated sanitization.

11.2. SSS: Harmonised high-level description

For sanitizable signature schemes (SSS) this thesis follows the initial description and desirable security properties given by *Ateniese, Chou, de Medeiros, and Tsudik* from 2005 [12]:

> "We define a sanitizable signature scheme as a secure digital signature scheme that allows a semi-trusted censor to modify certain designated portions of the message and produce a valid signature of the resulting (legitimately modified) message with no interaction with the original signer." [12].

Further, this thesis follows the later description by *Brzuska et al.* from 2009 [64]:

> "Sanitizable signature schemes [...] allow a signer to delegate signature rights in a controlled way. Namely, the signer can determine parts of the message which a designated party, the sanitizer, can later modify but such that the authenticity and integrity of the remaining parts is still guaranteed. In particular, even the sanitizer should not be able to change inadmissible parts of the message and produce a valid signature for such illegitimate transformations." [64]

[675] The works [197] and [196] from *Fleischhacker, Krupp, Malavolta, Schneider, Schröder, and Simkin* cite the following published papers containing results from this thesis: n° 20 [387], n° 11 [68], n° 17 [69]; see Appendix A.

[676] Original's endnote has been adjusted to correctly point to the same entry found in the bibliography of this thesis.

[677] It was first published in the joint work with *C. Brzuska* and *K. Samelin* at EuroPKI 2013 [69] (see Appendix A publication n° 17).

[678] These results have been published in the joint work with *C. Brzuska* and *K. Samelin* at EuroPKI 2012 [68] (see Appendix A publication n° 11).

Henceforth, in an SSS, the Signer signs a message m, where m is decomposable into ℓ blocks, i.e., $m = m[1] \,||\, m[2] \,||\, \ldots \,||\, m[\ell]$. Each $m[i]$ is a block, where $\ell \in \mathbb{N}^+$ is the number of blocks and $||$ denotes a reversible concatenation. For each block $m[i]$, the Signer decides whether sanitization by a Sanitizer is admissible or not. Each block's admissibility is codified in ADM. The designated Sanitizer can then later modify each admissible block $m[i]$ by replacing it with an arbitrary string $m[i]' \in \{0,1\}^*$ and create a new message m'. The desired modification is codified in MOD. Thus, the modified message can be obtained as $m' = \mathsf{MOD}(m)$. For this modified m', the Sanitizer can derive a new signature σ' without interaction with the Signer but using his own secret $\mathsf{sk}_{\mathrm{san}}$ and the Sanit algorithm. The resulting message-signature pair (m', σ') verifies under $\mathsf{pk}_{\mathrm{sig}}$, as long as the Signer authorized the Sanitizer for this modification from m to m', i.e., $\mathsf{MOD} \in \mathsf{ADM}$. The workflow was shown in Fig. 17 on page 54.

In SSS sanitization is not a public operation. Any valid sanitization in an SSS requires a secret denoted as $\mathsf{sk}_{\mathrm{san}}$. Thus, the Sanitizer becomes a designated party that knows the required secret.

The distribution of that sanitization key $\mathsf{sk}_{\mathrm{san}}$ is out of scope of the actual algorithms. As a corner case this thesis has analysed the case of it being publicly known, which models the opposite of integrity termed contingency. As this is of separate interest it is discussed in detail in Sec. 18.8.

11.3. SSS: Algorithmic description

The following algorithmic description follows the one introduced by *Ateniese et al.* [12] and formalised by *Brzuska et al.* [64]. A sanitizable signature scheme in general consists of the following set of mathematically efficient algorithms — probabilistic polynomial runtime (PPT) is considered efficient. The exact implementation of these algorithms differs strongly between the various existing schemes, this section describes each algorithm on the level of its input and output. In line with this thesis's definition of entities (see Sec. 3.6), the entity Signer always denotes the original signer of a given message m and Sanitizer denotes an entity that is authorized to sanitize at least one block of this m. Note, this section uses an array-like notation (see Definition 36 in Sec. 3.2) to index the blocks of a message for SSSs without loss of generality. How a structured, unstructured or partly structured message is decomposed into blocks must be described by each SSS instantiation individually.

The algorithms below incorporate the fixes proposed in the work of *Gong et al.* [216]: To prohibit the "rights forge attack" [216] the definitions of unforgeability, immutability, Signer-accountability and Sanitizer-accountability are amended such that the adversary will not win if it was able to maliciously modify the Signer's modification policy (ADM). In turn, this demands that ADM is recoverable from any valid message-signature pair (m, σ) if the oracle or algorithm requires it. This thesis will add ADM-recoverability as a correctness property for SSS in Sec. 11.4.4.

Definition 144 : Sanitizable Signature Scheme

SSS consists of at least seven efficient (PPT[679]) algorithms: SSS := (KGen$_{sig}$, KGen$_{san}$, Sign, Sanit, Verify, Proof, Judge).
Note, all algorithms may output $\perp$ in case of an error.

Key Generation: *There are two key generation algorithms, one for the Signer and one for the Sanitizer. Both create a pair of keys consisting of a private key and the corresponding public key:*

$$(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda), \qquad (\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda)\,.$$

The security parameter is denoted as λ.

Signing: *The Sign algorithm takes as input the security parameter, a message $m = (m[1], \ldots, m[\ell])$, with each block $m[i] \in \{0,1\}^*$, the Signer's secret key sk_{sig}, the Sanitizer's public key pk_{san}, as well as a description ADM of the admissibly modifiable blocks. It is assumed that ADM is always recoverable from a signature σ valid under pk_{sig} by the Signer holding the corresponding secret key sk_{sig}. It outputs a signature σ and the message (or $\perp$ on error):*

$$(m, \sigma) \leftarrow \mathsf{Sign}(1^\lambda, m, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM})\,.$$

[679] Probabilistic polynomial-time (PPT); see Definition 11 for definition.

Sanitizing: *The algorithm* Sanit *takes the security parameter, a message* $m = (m[1], \ldots, m[\ell])$*, with each block* $m[i] \in \{0,1\}^*$*, a modification instruction* MOD*, a signature* σ*, the* Signer*'s public key* pk_{sig} *and the* Sanitizer*'s secret key* sk_{san}*. It modifies the message* m *according to the modification instruction* MOD*, which contains pairs* $(i, m[i]')$ *for those blocks that shall be modified. It is assumed that* ADM *is always recoverable from a signature* σ *endorsing* pk_{san} *by the* Sanitizer *holding the corresponding secret key* sk_{san}*.* Sanit *generates a new signature* σ' *for the modified message* $m' \leftarrow \mathsf{MOD}(m)$*.* Sanit *outputs* m' *and* σ' *(or* $\perp$ *in case of an error):*

$$(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})\,.$$

Verification: *The* Verify *algorithm takes the security parameter , a message* $m = (m[1], \ldots, m[\ell])$*, with each block* $m[i] \in \{0,1\}^*$*, the signature* σ*, and the public keys* pk_{sig} *and* pk_{san} *as input. It is assumed that* ADM *is recoverable from any signature* $\sigma \neq \perp$ *unless the* Verifier *is not authorized, i.e.* ADM *is assumed recoverable for* $\geq$ *BFD integrity*[680]*.* Verify *outputs a decision* $d \in \{\texttt{true}, \texttt{false}\}$ *indicating the validity of the signature* σ *for the message* m *with respect to the public keys* pk_{sig} *and* pk_{san}*:*

$$d \leftarrow \mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san})\,.$$

Proof: *The* Proof *algorithm takes as input the security parameter, the* Signer*'s secret key* sk_{sig}*, a message* $m = (m[1], \ldots, m[\ell])$*, with each block* $m[i] \in \{0,1\}^*$*, and a signature* σ *as well a set of (polynomially many) additional message-signature pairs* $(m_i, \sigma_i)_{i=1,2,\ldots,k}$ *and the public key* pk_{san}*. It is assumed that* ADM *is always recoverable from a signature* σ *valid under* pk_{sig} *by the* Signer *holding the corresponding secret key* sk_{sig}*.* Proof *outputs a string* $\pi \in \{0,1\}^*$ *(or* $\perp$ *in case of an error):*

$$\pi \leftarrow \mathsf{Proof}(1^\lambda, \mathsf{sk}_{sig}, m, \sigma, (m_1, \sigma_1), \ldots, (m_k, \sigma_k), \mathsf{pk}_{san})\,.$$

Judge: *Algorithm* Judge *takes as input the security parameter, a message* $m = (m[1], \ldots, m[\ell])$*, with each block* $m[i] \in \{0,1\}^*$ *and a valid signature* σ*, the public keys of the parties* pk_{sig} *and* pk_{san} *and a proof* π*. It is assumed that* ADM *is always recoverable from a signature* σ *valid under* pk_{sig} *by the* Arbitrator *from* π *created genuinely by the* Signer *that corresponds to* pk_{sig}*.* Judge *outputs a decision* $d \in \{\texttt{Sig}, \texttt{San}\}$ *indicating whether the message-signature pair has been created by the* Signer *or the* Sanitizer *(or* $\perp$ *in case of an error):*

$$d \leftarrow \mathsf{Judge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi)\,.$$

11.4. SSS: Correctness properties

This thesis assumes that an SSS is correct. Namely, that

- all genuinely signed messages (m, σ) are accepted as valid by Verify,
- all genuinely sanitized messages (m, σ) are accepted as valid by Verify,
- every genuinely created proof by the Signer leads the Arbitrator to decide in the Signer's favour, and
- the modification policy (ADM) can be uniquely and efficiently recovered from a valid messages-signature pair by authorized entities.

[680] Recall from Sec. 6.3.2: BFD level means that for all individual blocks, the potential future change can be detected. This means that the Verifier is allowed to find out which blocks are admissible for modifications and which blocks are not. Thus, leaking ADM to the Verifier through a valid message-signature pair is not a problem. See also Sec. 11.4.4 for more details.

That is, the formally defined correctness properties *signing correctness, sanitizing correctness* and *proof correctness* as presented by *Brzuska et al.* [64, 68] hold. These are restated in the thesis's notation in the following. Additionally, this thesis presents the formal definition for ADM recovery.

11.4.1. Signing correctness for SSS

Definition 145 : Signing correctness (from[681] *Brzuska et al.* [64])

For any security parameter λ,
any key pair $(\mathsf{sk}_{sig}, \mathsf{pk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda)$,
any key pair $(\mathsf{sk}_{san}, \mathsf{pk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda)$,
any message $m \in \{0,1\}^*$,
any ADM,
and any $\sigma \leftarrow \mathsf{Sign}(1^\lambda, m, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM})$
it is required that:

$$\mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}.$$

11.4.2. Sanitizing correctness for SSS

Definition 146 : Sanitizing correctness (from[681] *Brzuska et al.* [64])

For any security parameter λ,
any key pair $(\mathsf{sk}_{sig}, \mathsf{pk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda)$,
any key pair $(\mathsf{sk}_{san}, \mathsf{pk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda)$,
any message $m \in \{0,1\}^*$,
any σ *with* $\mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}$,
any $\mathsf{MOD} \subseteq \mathsf{ADM}$ *where* ADM *is reconstructed from* (m, σ),
and any pair $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})$
it is required that:

$$\mathsf{Verify}(1^\lambda, m', \sigma', \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}.$$

11.4.3. Proof correctness of SSS

Definition 147 : Proof correctness (from[681] *Brzuska et al.* [64])

For any security parameter λ,
any key pair $(\mathsf{sk}_{sig}, \mathsf{pk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda)$,
any key pair $(\mathsf{sk}_{san}, \mathsf{pk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda)$,
any message $m \in \{0,1\}^*$,
any signature σ,
any $\mathsf{MOD} \subseteq \mathsf{ADM}$ *where* ADM *is reconstructed from* (m, σ),
any $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})$
with $\mathsf{Verify}(1^\lambda, m', \sigma', \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}$,
any (polynomially many) $m_1, \ldots, m_q$ *and* $\mathsf{ADM}_1, \ldots, \mathsf{ADM}_q$
with $\sigma_i \leftarrow \mathsf{Sign}(1^\lambda, m_i, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM}_i)$ *and*
$(m, \sigma) = (m_i, \sigma_i)$ *for some i*,
and any $\pi \leftarrow \mathsf{Proof}(1^\lambda, \mathsf{sk}_{sig}, m', \sigma', m_1, \sigma_1, \ldots, m_q, \sigma_q, \mathsf{pk}_{san})$

it is required that:

$$\mathsf{Judge}(m', \sigma', \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi) = \texttt{San}.$$

[681] Original notation has been adjusted to this thesis's notation.

11.4.4. ADM recovery correctness and ease of decomposition for SSS

The work of *Gong et al.* requires that every SSS allows ADM to be correctly recoverable when needed [216]. Care has been taken to not generally require that all parties must be able to recover the original ADM as this might lead to unwanted losses of confidentiality of what can be redacted. The Verifier shall be able to verify on a valid message-signature pair (m, σ) that ADM was not breached, but must not always learn ADM's contents. Especially the Verifier shall not be able to recover ADM from the message-signature if the integrity protection aims to offer the Verifier with only a limited detection capability for future changes, i.e., the levels NFD or 1FD, as discussed in Sec. 6.3.2. As $\geq$ 1FD integrity protection prohibits to reconstruct ADM from just a valid message-signature pair (m, σ) it requires additional secrets for the parties (Signer, Sanitizer, Arbitrator). The Signer holding the secret key sk_{sig} shall be able to recover ADM from a message-signature pair (m, σ) that is valid under his corresponding pk_{sig}. A Sanitizer might require its secret key sk_{san} that corresponds to pk_{san} endorsed by the Signer to recover ADM. Finally, the Arbitrator might need the Signer-created proof π to recover ADM.

Definition 148 : ADM recovery correctness

> *For any genuinely created message-signature pair (m, σ) for which $\mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) =$ `true`, it is required that:*
>
> *ADM can be correctly and efficiently (PPT) reconstructed by authorized entities.*

Another assumption is that the decomposition of the message into blocks is efficiently possible for any algorithm that requires this. This means that every genuinely created message-signature pair (m, σ) allows to be efficiently (PPT) and correctly decomposed into all blocks of m. This follows from the concatenation of blocks being reversible.

Definition 149 : ease and correctness of decomposition

> *For any genuinely created message-signature pair (m, σ) for which $\mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) =$ `true`, and all $k \leq |m|$ where $|m|$ denotes the number of blocks in m it is required that:*
>
> $$m'[k] \leftarrow \mathsf{decompose}(k, m) \textit{ is PPT efficient and } m'[k] = m[k] \textit{ if } m[k] \in m\ .$$

11.5. SSS: Sequential executions of Sanit

The sequential execution of the Sanit algorithm on the same message may result in an admissible block being subject to several consecutive modifications. After the sequential execution, a valid sanitized message-signature pair is affected by the sum of all applied modifications. In a private scheme any modification of a block cloaks all previous modifications: The application of the Sanit algorithm on a validly sanitized message-signature pair has the same cloaking effect towards the confidentiality of the already sanitized contents as would the first application of the Sanit algorithm that cloaks the original contents of that block. This is in line with the expectation of this subsequent modification being not done for version control, but as a privacy preserving action.

Further, the capability of Sanit with respect to the upper bound of authorized modifications does not change by executing it sequentially, e.g., if a certain block is restricted and cannot initially be modified, then this will not change even under subsequent executions of the Sanit algorithm on that message.

For example assume the following: Let $m = m_1||m_2||m_3$ be signed with a secure SSS, i.e., especially the SSS needs to be private, unforgeable, immutable. Further let m_1 be not admissible, i.e., $m[\mathsf{fix}] = 1$. Starting from $m = A||B||C$, a first valid sanitization could result in $m' = A||D||E$. An additional second successful sanitization of m_3 to F would then result in a validly signed $m'' = A||D||F$. A Verifier receiving m'' and the accompanying sanitized signature will only see $A||D||F$, as the scheme is private. m_1 remains non admissible. EXAMPLE

Likewise, Sanitizer- or Signer accountability will hide previous sanitizations and appoint the last Sanitizer as accountable. Definition 150 captures this as follows:

Definition 150 : Sequential executions of Sanit for SSS

For any security parameter λ,
any key pair $(\mathsf{sk}_{sig}, \mathsf{pk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda)$,
any key pair $(\mathsf{sk}_{san}, \mathsf{pk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda)$,
any message $m \in \{0,1\}^*$ *with* ℓ *blocks,*
any σ *with* $\mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}$,
any MOD_1 *matching* ADM *from* σ,
any pair $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}_1, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})$ *with* $\mathsf{Verify}(1^\lambda, m', \sigma', \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}$,
any pair $(m'', \sigma'') \leftarrow \mathsf{Sanit}(1^\lambda, m', \mathsf{MOD}_2, \sigma', \mathsf{pk}_{sig}, \mathsf{sk}_{san})$ *with* $\mathsf{Verify}(1^\lambda, m'', \sigma'', \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}$
and any $m[i]$ *where* MOD_2 *matches* ADM *from* (m', σ'),
any pair $(m''', \sigma''') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}_2, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})$ *with* $\mathsf{Verify}(1^\lambda, m''', \sigma''', \mathsf{pk}_{sig}, \mathsf{pk}_{san}) = \texttt{true}$
and any $m[i]$ *where* MOD_2 *matches* ADM *from* (m, σ),

it is required that:

$$\forall i, 1 \leq i \leq \ell \text{ and } i \in \mathsf{MOD}_2 : m[i]'' = m[i]''' .$$

Note that Definition 150 holds also if a modifiable block i has previously already been modified, i.e., it holds also if $\{i | i \in \mathsf{MOD}_1\} \cap \{j | j \in \mathsf{MOD}_2\} \notin \emptyset$. However, in each block of a validly sanitized message reflects only the last modification, i.e., the order of sanitizations may matter. Further, even in block-level non-interactive publicly accountable schemes[682] it is not detectable if a block was subject to previous subsequent modifications or was only once sanitized from the original[683].

This is in line with the expected behaviour of subsequent overlapping authorized modifications to a document: *Schneier* stated that “[t]he best definition [for integrity] I’ve seen is: ‘Every piece of data is as the last authorized modifier left it.‘ ” [433, p. 122][684].

11.6. SSS: Existing security properties in harmonised notation

This section gives the state-of-the-art security properties for SSS in the notation used in this thesis. The cryptographic properties are as follows:

- unforgeability (Sec. 11.6.1),
- immutability (Sec. 11.6.2),
- standard privacy (Sec. 11.6.3),
- transparency (Sec. 11.6.4),
- unlinkability (Sec. 11.6.5), and
- Signer- and Sanitizer-accountability (Sec. 11.6.6).

11.6.1. Unforgeability for SSS

This property got introduced by *Ateniese et al.* in [12] and based on *Brzuska et al.*’s [64] formalisation, this thesis defines unforgeability. *Brzuska et al.* already noted that unforgeability for SSS is close to the notion of unforgeability of classic digital signature schemes [64].

Definition 151 : Unforgeability for SSS (high-level)

Unforgeability *guarantees that a signature valid under* pk_{sig} *on a message can only be generated*

- *by the Signer with access to* sk_{sig} *corresponding to* pk_{sig}, *or*

[682] See Sec. 6.4.3 for an introduction to this property and Sec. 6.5 for one on 'block-level'; see Definition 190 for the properties definition for SSS in Sec. 13.2.

[683] In transparent schemes it is not even detectable if a block was subject to a subsequent modification at all (see Sec. 3.10.1 for an introductory discussion; see Definition 158 for the property definition for SSS in Sec. 11.6.4).

[684] Unfortunately, *Schneier* did not provide the source of this definition of integrity in [433].

- *by a* Signer-*endorsed* Sanitizer *with access to* sk_{san} *that is sanitizing a message-signature pair where the signature is valid under* pk_{sig} *and where all sanitizations are conforming to the* Signer-*endorsed modification policy (*ADM*).*

Note 1 All valid sanitizations are excluded from being forgeries.

Note 2 Unforgeability is as strong as possible; it shall explicitly prohibit unauthorized re-ordering of blocks of structured data, as well as mix-and-match attacks.

The following formal definition already incorporates the criticism of *Gong et al.* [216]. *Gong et al.* [216] first showed this weakness for *Brzuska et al.*'s security model. To avoid the problem described by *Gong et al.* [216], the suggested fix has been applied for the property of unforgeability: A maliciously forged ADM, denoted as ADM*, will also lead to the success of the adversary. This thesis has raised this already to a correctness property for SSS in Sec. 11.4.4.

Definition 152 : Unforgeability for SSS

An SSS *is* ***unforgeable****, if for any efficient (PPT) adversary* $\mathcal{A}$ *the probability that the experiment* $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ *depicted in Fig. 47 returns* 1*, is negligible (as a function of* λ*).*

```
**Experiment** $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}) \leftarrow \mathsf{KGen_{sig}}(1^\lambda)$
  $(\mathsf{pk_{san}}, \mathsf{sk_{san}}) \leftarrow \mathsf{KGen_{san}}(1^\lambda)$
  $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \mathsf{sk_{sig}}, \cdots)\mathsf{Proof}(1^\lambda, \mathsf{sk_{sig}}, \cdots), \mathsf{Sanit}(1^\lambda, \cdots, \mathsf{sk_{san}})}(1^\lambda, \mathsf{pk_{sig}}, \mathsf{pk_{san}})$
    for $i = 1, 2, \ldots q$ let $(m_i, \mathsf{ADM}_i, \mathsf{pk_{san,i}})$ and $\sigma_i$
      denote the adaptive queries and answers to and from the oracle Sign
    for $j = q+1, \ldots, r$ let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk_{sig,j}})$ and $(m'_j, \sigma'_j)$
      denote the adaptive queries and answers to and from the oracle Sanit
  return 1, if
    $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk_{sig}}, \mathsf{pk_{san}}) = \mathtt{true}$ and
    $\forall i = 1, 2, \ldots, q : (\mathsf{pk_{san}}, m^*, \mathsf{ADM}^*) = (\mathsf{pk_{san,i}}, m_i, \mathsf{ADM}_i)$ and
    $\forall j = q+1, \ldots, r : (\mathsf{pk_{sig}}, m^*, \mathsf{ADM}^*) = (\mathsf{pk_{sig,j}}, m^*, \mathsf{ADM}_j)$
```

Figure 47. Unforgeability Experiment for SSS based on *Brzuska et al.* [64] and *Gong et al.* [216]

In the unforgeability game depicted in Fig. 47 the adversary is not in the possession of any of the two secret keys $\mathsf{sk_{sig}}$ and $\mathsf{sk_{san}}$. Both keys are fixed in the beginning and are kept within the respective oracles. The adversary can generate genuinely signed or sanitized message-signature pairs for messages of its own choosing by accessing the oracles Sign and Sanit. Further, the adversary can obtain genuinely generated proofs by accessing the oracle Proof. The adversary wins, e.g., the game outputs 1, if the adversary algorithm $\mathcal{A}$ can create a *fresh* message m^* with a valid signature σ^* verifying under the public keys corresponding to above fixed secret keys. Here, fresh means that the message or the corresponding ADM was not generated by a query to neither oracle involving the above fixed keys. Note, due to correctness the adversary can reconstruct ADM from any valid message-signature pair, but it could also output a message-signature pair (m^*, σ^*) which contains a crafted ADM* which breaks unforgeability.

11.6.2. Immutability for SSS

This property got introduced by *Ateniese et al.* in [12]. Based on *Brzuska et al.* [64], this thesis defines immutability as follows:

Definition 153 : Immutability for SSS (high-level)

Immutability *prevents a* Sanitizer *from deriving a verifying signature after modifying any block(s) not marked as admissible for this* Sanitizer *by the* Signer*.*

The following formal definition incorporates the criticism of *Gong et al.* [216]. For immutability *Gong et al.* proved that in *Brzuska et al.*'s security model it is additionally necessary to cover the description of the admissible parts inside the formal game [216].

Definition 154 : Immutability for SSS

> *An SSS is immutable, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment $\mathsf{Immutability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 48 returns 1 is negligible (as a function of λ).*

```
**Experiment** $\mathsf{Immutability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $\mathcal{Q} \leftarrow \emptyset$
  $(m^*, \sigma^*, \mathsf{pk}^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathsf{sig}},\cdots),\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathsf{sig}},\cdots)}(1^\lambda, \mathsf{pk}_{\mathsf{sig}})$
  let $(m_i, \mathsf{ADM}_i, \mathsf{pk}_{\mathsf{san,i}})$ for $i = 1, \ldots, q$ be queries to Sign
  For each query to Sign, do:
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mathsf{ADM}_i, \mathsf{pk}_{\mathsf{san,i}}, m_i') \mid m_i' \in \{\mathsf{ADM}_i(m_i)\}\}$
  return 1, if:
    $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathsf{sig}}, \mathsf{pk}^*) = \mathtt{true}$ and
    $(\mathsf{ADM}^*, \mathsf{pk}^*, m^*) \notin \mathcal{Q}$
```

Figure 48. Immutability Experiment for SSS; based on *Brzuska et al.* [64] and *Gong et al.* [216]

The adversary gets access to a signing oracle. The signing key pair for the oracle is generated accordingly and the adversary has the ability to retrieve valid signatures for adaptively chosen messages for a Sanitizer's public key of the adversary's choice. Any proper sanitization of a message that was signed and where this sanitization for that Sanitizer was endorsed by the signing oracle is not considered a forgery. The adversary, even in the role of a Sanitizer, must only be able to alter *admissible* block(s). Namely, it must not alter the content, structure or relations of immutable block(s) if the scheme protects that block's structural or relational integrity, must not remove block(s), and must not append a block. Finally, the adversary must not be able to swap the endorsed Sanitizer's public key, nor to modify the list of admissible blocks (ADM). The latter was explicitly stated by *Gong et al.* [216][685].

To capture this formally, the game describing the notion must remember for all queried messages the structural position and the content of immutable blocks (s). The game depicted in Fig. 48 in the line $\{(\mathsf{ADM}_i, \mathsf{pk}_{\mathsf{san,i}}, m_i' \mid m_i' \in \{\mathsf{ADM}_i(m_i)\}\}$ uses a set-builder notation to build 3-tuples for all potential sanitized messages that are possible for the queried message under the queried modification policy ADM_i. The 3-tuple takes additional note of the queried policy ADM_i and the queried sanitizer key $\mathsf{pk}_{\mathsf{san,i}}$. The representation in the game is very short as by notation the set denoted $\{\mathsf{ADM}_i(m_i)\}$ contains all possible modifications of m_i that can be done under policy ADM_i[686]. In total, the set of all proper sanitizations of all oracle queries ever done by the adversary is collected in $\mathcal{Q}$.

As in the game presented originally by *Brzuska et al.* [64] the attacker has two options for an attack on immutability (1) exchange the endorsed Sanitizer key or (2) modify a block originally marked as not admissible. Moreover, the adversary (3) could manipulate the original policy ADM in order to gain modifiability; this needs to be also prohibited as required by *Gong et al.* [216]. All three possibilities for an attack on immutability are captured within the second winning requirements that states that $(\mathsf{ADM}^*, \mathsf{pk}^*, m^*) \notin \mathcal{Q}$. Note, the above makes ADM explicit, even though it is stated to be uniquely recoverable from a valid message-signature pair. Hence, ADM^* does not need to be provided by the adversary algorithm, but is reconstructed from the crafted valid message-signature pair (m^*, σ^*). This message-signature pair then breaks immutability due to the crafted ADM^* that is different to the endorsed list of admissible blocks.

When comparing the original game from *Brzuska et al.* [64] given in Fig. 49 it should be noted that in order to break immutability the adversary has to provide a modified message m^* that was not endorsed by the Signer, i.e. is an unauthorized modification of an already signed triple of $m_i, \mathsf{ADM}_i, \mathsf{pk}_{\mathsf{san,i}}$. This means that immutability is not concerned with forgeries in general or if the adversary can change the

[685] "Hence any modification of ADM should also be regarded as a malicious behavior and captured by the security model." [216].

[686] See this thesis' notation as described in Definition 59.

```
Experiment $\mathsf{Immutability-Brzuska-et-al.}^{SanSig}_{\mathcal{A}}(n)$
  $(\mathsf{pk_{sig}}, \mathsf{sk_{sig}}) \leftarrow \mathsf{KeyGen}(1^n)$
  $(m^*, \sigma^*, \mathsf{pk}^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk_{sig}}, \cdots), \mathsf{Proof}(1^\lambda, \mathsf{sk_{sig}}, \cdots)}(\mathsf{pk_{sig}})$
    letting $(m_i, \mathsf{ADM}_i, \mathsf{pk_{san,i}})$ and $\sigma_i$ for $i = 1, \ldots, q$
    denote the queries and answers to and from oracle Sign.
  return 1, if
    $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk_{sig}}, \mathsf{pk}^*) = \mathtt{true}$ and
    for all $i = 1, 2, ..., q$ we have
      $\mathsf{pk}^*_{\mathsf{san}} \neq \mathsf{pk_{san,i}}$, or
      $m^*[j_i] \neq m_i[j_i]$ for some $j_i \notin \mathsf{ADM}_i$
      //where shorter messages are padded with blocks of the special symbol $\perp \notin \{0,1\}^*$
```

Figure 49. Immutability Experiment for SSS as originally stated by *Brzuska et al.* [64]

```
Experiment $\mathsf{Immutability-Camenisch-et-al}^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{sk_{sig}}, \mathsf{pk_{sig}}) \leftarrow \mathsf{KGen_{sig}}(1^\lambda)$
  $(m^*, \sigma^*, \mathsf{pk}^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk_{sig}}, \cdots), \mathsf{Proof}(1^\lambda, \mathsf{sk_{sig}}, \cdots)}(1^\lambda, \mathsf{pk_{sig}})$
    for $i = 1, 2, \ldots, q$ let $(m_i, \mathsf{pk}_{\mathsf{san},i}, \mathsf{ADM}_i)$ index the queries to Sign
  return 1, if $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk_{sig}}, \mathsf{pk}^*) = \mathtt{true} \wedge$
    $(\forall i \in \{1, 2, \ldots, q\} : \mathsf{pk}^* \neq \mathsf{pk}_{\mathsf{san},i} \vee$
      $m^* \notin \mathsf{span}_\vdash(m_i, \mathsf{ADM}_i))$
  return 0
```

Figure 50. Immutability Experiment for SSS revised; as jointly published in *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107] and *Beck, Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [32]; notationally adjusted to this thesis notation

message under a $\mathsf{pk_{san}}$ that was never endorsed, i.e. never queried to the oracle Sign. To match the high-level goal the winning condition from *Brzuska et al.*'s game in [64] must evaluate both sub-conditions at once for each query i to the oracle. With explicit brackets[687] it would read — after notational adjustments[688] — as given in Fig. 50.

This thesis will use the notion of "immutability" to mean either the interpretation given in Fig. 50 [107] or the interpretation with the set-builder given in Fig. 48 [32, 107].

11.6.3. Standard Privacy for SSS following *Brzuska et al.*

Privacy is a core security property of SSS, and required as per Requirement 9. It has been described in [12] and formally described first by *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk* [64]. Here the thesis will not re-state the discussion about the original definition that was discussed in Sec. 9.1.5.1. Note that the following privacy definition according to *Brzuska et al.* [64] only considers outsiders as privacy adversaries, e.g., adversary has no knowledge of any private key(s)[689].

[687] The brackets are also introduced into recently published definitions of joint work: It can be found in *Bilzhause, Huber, Pöhls, and Samelin* [48] as published in ARES 2016. It is captured in the latest update of the full version of the joint work by *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107] published at the PKC 2017 conference (updated 2017-12-13), see `https://eprint.iacr.org/2017/011/20171213:114927` [last accessed: Jan. 2018]. It is also captured in a version containing the brackets in the joint work by *Beck, Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [32] published at the ACISP 2017 conference (see Appendix A publication nº 57).

[688] The original used "'$m^* \notin \{\mathsf{MOD}(m_i) \mid \mathsf{MOD}$ with $\mathsf{MOD}(\mathsf{ADM}_i) = \mathtt{true}\}$)" to denote that the m^* shall be a message that does not fit any derivation that can be reached within the authorized modifications ADM_i, i.e. was intended to be unauthorized by the Signer but due to the successful attack verifies under the Signer's public key. This got notationally adjusted using the thesis notation of $\mathsf{span}_\vdash(m_i, \mathsf{ADM}_i)$ which denotes all messages that have been authorized by ADM_i to be derivable from m_i with a valid signature.

[689] As an extension, this thesis describes strengthened privacy in Sec. 15.1.3.

Definition 155 : Standard Privacy

> *A **private** scheme prevents* Verifiers *from recovering any information (esp. the original value) about block(s) of m, which are no longer in the sanitized block(s) of m', through a valid signature σ' over m'.*
>
> *Note: Information leakage through the semantic content of the modified message m' itself, which is given to the adversary, is out of scope.*

More details regarding the note on no leakage through the message see the discussion in Sec. 9.4.3. The exclusion of leakage on the semantic level is adopted by the thesis and is also found in other original papers on MSS[690]. It was adopted as it allows the security properties, and thus the mechanism, to work without the need to semantically understand the message's content they protect. Leakage through the message is a long standing problem in redaction or sanitization of content. In any case it must be detected and controlled by the human user. It might be easier to spot than information leaking through the adjusted cryptographic signature of the sanitized or redacted message. The analysis of legal texts on digital sanitization showed that it is up to the application to carry out all necessary subsequent modifications to achieve the desired confidentiality on the semantic level: for example the guidance provided for correct redactions or sanitizations when preparing answers to freedom of information requests as previously discussed in Sec. 8.1.

A new property refinement given by this thesis distinguishes between *standard privacy* and *strengthened privacy*, this is discussed in Sec. 15.1.3 of this thesis[691]. This thesis will henceforth call this *standard privacy* according to *Brzuska et al.*:

Definition 156 : Standard Privacy for SSS following *Brzuska et al.*

> *An SSS is **private**, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 51 returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ).*

Definition 156 follows from the original one given by *Brzuska et al.* [64], and the idea is based on the indistinguishability notion for encryption, i.e., the adversary shall not be able to differentiate the used input to the cryptographically private operation of a sanitization.

In more detail, in the privacy game depicted in Fig. 51, the adversary is able to choose the original message. Additionally, the adversary can use signature, sanitize and proof oracles to generate and verify signed and sanitized messages. Hence, the adversary is not given access to any secrets, i.e., the adversary does not know $\mathsf{sk}_{\mathsf{sig}}$ and $\mathsf{sk}_{\mathsf{san}}$. However, it can access the LoRSanit oracle, for which the basic idea is as follows: The oracle randomly either first signs and sanitizes sanitizes the first message (m_0) or the second (m_1). The oracle will only do so if the resulting message will be the same, i.e., m' and ADM_i will not give the adversary any additional hints. The five abort conditions check this. After those conditions checked, it yields that the only information left to adversary to distinguish the two outputs is the signature.

The adversary must decide which message was used as input. In other words, in the model a breach of privacy occurs if the adversary provides two crafted input messages and modification instructions to the LoRSanit oracle and later is able to decide which one was randomly chosen and used by the oracle to produce the signed outcome with a probability better than $\frac{1}{2}$. Of course trivial ways to win the game are excluded. This is described with the five abort conditions: First it is checked if both modifications of the supplied messages would result in the same sanitized message ($\mathsf{MOD}_0(m_0) = \mathsf{MOD}_1(m_1)$). The second and third condition exclude cases where the outputted sanitized message would differ due to errors induced due to specially crafted input messages and modification instructions or due to different ADMs. The fourth is due to the recent discussion of *Krenn et al.* [294] to make sure that the supplied policies ($\mathsf{ADM}_{\{0,1\}}$) before and after modification match the respective message. The final one ($\mathsf{MOD}_0(\mathsf{ADM}_0) \neq \mathsf{MOD}_1(\mathsf{ADM}_1)$) means that — if a modification to ADM is allowed by

[690] "Note that we do not consider semantics of the document." [349]; "As information leakage through the modified message itself can never be prevented, we only refer to information which is available through the sanitized signature." [67].

[691] Note, the strengthened privacy notion is a contribution of this thesis.

the scheme — the ADM' of the resulting sanitized message m' shall be the same regardless what input message was used for the sanitization. This was not found in the original game in [66]. It is added to cater for schemes that would offer consecutive sanitization control (see Sec. 14.5.1.1), which would be an authorized modification of ADM.

Note, in standard privacy the adversary is not able to supply any signatures for the input messages itself and does not know any secret keys, neither sk_{san} nor sk_{sig}. In standard privacy the adversary is able to define the messages that are input to the LoRSanit oracle and by specifying modification instructions the adversary can influence the result that shall be generated by the sanitization. However, the inner workings, in particular the signatures σ_i and message m_i, are not given to the adversary [64].

```
Experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\text{sig}}, \mathsf{sk}_{\text{sig}}) \leftarrow \mathsf{KGen}_{\text{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\text{san}}, \mathsf{pk}_{\text{san}}) \leftarrow \mathsf{KGen}_{\text{san}}(1^\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk}_{\text{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\text{san}})}_{\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\text{sig}},\cdots),\mathsf{LoRSanit}(1^\lambda,\ldots,\mathsf{sk}_{\text{sig}},\mathsf{sk}_{\text{san}},b)}(1^\lambda, \mathsf{pk}_{\text{sig}}, \mathsf{pk}_{\text{san}})$
    where oracle LoRSanit on input of $m_0, \mathsf{MOD}_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$ // resulting $m'$ is the same
      if $\mathsf{MOD}_0 \nsubseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{MOD}_1 \nsubseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{ADM} \nsubseteq m_0$ or $\mathsf{ADM} \nsubseteq m_1$ or $\mathsf{MOD}_0(\mathsf{ADM}) \nsubseteq \mathsf{MOD}_0(m_0)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}) \neq \mathsf{MOD}_1(\mathsf{ADM})$, return $\perp$
      let $(m_i, \sigma_i) \leftarrow \mathsf{Sign}(1^\lambda, m_b, \mathsf{sk}_{\text{sig}}, \mathsf{pk}_{\text{san}}, \mathsf{ADM})$
      return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_i, \mathsf{MOD}_b, \sigma, \mathsf{pk}_{\text{sig}}, \mathsf{sk}_{\text{san}})$
  return 1, if $a = b$
```

Figure 51. Standard Privacy Experiment for SSS based on *Brzuska et al.* [64] including additions from *Krenn et al.* [294]

11.6.4. Transparency for SSS

The notion of transparency as prevailing in SSS literature was introduced by *Ateniese, Chou, de Medeiros, and Tsudik* in two "flavors" [12][692] with two different scopes. The following analysis yields that their definition of the notion must be revised[693].

Ateniese et al. [12] define the property of transparency (T) as follows:

> "Given a signed message with a valid signature, no party — except the censor and the signer — should be able to correctly guess whether the message has been sanitized" [12].

They further divide the property into *weak transparency* (WT) and *strong transparency* (ST):

> "We further distinguish among two flavors of transparency: weak and strong. Weak transparency means that the verifier knows exactly which parts of the message are potentially sanitizable and, consequently, which parts are immutable. In contrast, strong transparency guarantees that the verifier does not know which parts of the message are immutable and thus does not know which parts of a signed message could potentially be sanitizable."[692] [12]

Essentially, T always implies exactly one of WT or ST and vice versa WT or ST imply T:

$$\Big((T \Rightarrow \ (ST \dot{\vee} WT)) \wedge ((ST \dot{\vee} WT) \ \Rightarrow \ T) \Big) \equiv \Big(T \Leftrightarrow (ST \dot{\vee} WT) \Big) .$$

[692] The American English spelling from original work is retained for terms and quotes.

[693] This result was published as part of the joint paper by *Pöhls, Samelin, and Posegga* at ACNS 2011 [391] (see Appendix A publication n° 6).

In the above $\dot{\vee}$ denotes an 'exclusive or'. Practically, a Verifier either knows which m_i is potentially sanitizable or it does not. Hence, either ST or WT is provided by a scheme. With the way *Ateniese et al.* defined those notions it should be a tautology, i.e., $\models (ST \dot{\vee} WT)$. Since $T \Leftrightarrow (ST \dot{\vee} WT)$ this results that T is always true for any scheme. This is counterintuitive and contradicts SSS constructions from MSS literature, such as the one described by *Miyazaki et al.* [347].

This lack of block-level properties is identified as Shortcoming 4 (see Sec. 12.4) and will be rectified by a new construction given in this thesis named $blockaccSSS$ (see Subsections 13.4.1 and 13.4.3).

11.6.4.1. Definition of transparency on the scope of message-signature pairs

In order to define the property of transparency, this analysis starts by stating the property in natural language along what has been stated in other works.

The work of *Brzuska et al.* [64] offers an example why one wants to hide that "[...] the sanitizer inherits the signing authority. Sometimes knowledge of this fact makes the sanitized data less valuable, e.g., an original business plan coming from the CEO is a more desirable target for a spy than the sanitized plan from the administration office. Transparency now says that no one except for the signer and the sanitizer should be able to distinguish signatures from the signer and the sanitizer." [64]. In contrast, *Ateniese et al.* [12] — even-though they also define the term — give an example why this property might not be useful: "[...] if a document originally signed by some government official is later released by a certain government agency -- acting as a censor -- under the Freedom of Information Act, the general public would likely prefer knowing which parts of the document could have been sanitized." [12].

In line with the existing works this is captures as follows in this thesis:

Definition 157 : Cryptographic Transparency

> ***Transparency** prevents third parties which have only access to a given valid message-signature pair (m, σ) and public information (e.g., pk_{sig} or pk_{san}) to decide which party (Signer or Sanitizer) is accountable for a given valid message-signature pair (m, σ).*
>
> *Notes for Definition 157:*
>
> *Note 1 Information leakage through the message itself is out of scope of Definition 157.*
>
> *Note 2 Transparency is cryptographically stronger than the cryptographic privacy notion (see Definition 155), i.e., transparency implies privacy [64].*
>
> *Note 3 Third party refers here to any party that is neither Signer or Sanitizer, i.e. an entity in the role of the Verifier. The third party has no access to non-public information; especially not to the secret keys (sk_{sig} or sk_{san}) or output generated involving them, i.e. no access to the proof π for the given valid message-signature pair (m, σ).*

Transparency is not mandatory for SSS: Transparency is a stronger privacy property, as transparency implies privacy [64]. Transparency is important, if the existence of a Sanitizer must be hidden to increase the privacy. This is required, if sanitization leads to disadvantages of any party involved [64]. Still, *Brzuska et al.* do not consider the property of transparency to be mandatory for a meaningful SSS [64]. Transparency would hinder the detection of subsequent modifications by Verifiers alone; this violates the recommendation against public accountability stated Requirement 4.1[a]. In line with the goal of this thesis to increase the probative value this thesis will trade cryptographic transparency for the property of non-interactive public accountability. Dropping transparency is more favourable from a legal perspective as the analysis of the impact on legal certainty highlights which was presented in Subsections 6.4.2 and 6.4.5.

[a] Requirement 4.1: Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer.

Brzuska et al. formalised the property of transparency as follows:

Definition 158 : Proof-Restricted Transparency for SSS following *Brzuska et al.*

> *An SSS is proof-restricted transparent, if for any efficient algorithm $\mathcal{A}$ the probability that the experiment $\mathsf{Transparency}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 53 returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ).*

In a transparent scheme the adversary as a Verifier is not able to decide whether the message-signature pair carries a signature created through Sanit or a freshly signed signature generated by Sign. This yields even if the adversary controls the input message.

A transparent MSS still offers the possibility to determine the accountability for a message by the execution of the Judge algorithm. The algorithm Judge requires additional input from the Proof algorithm. The adversary cannot run the Proof algorithm itself as it is not the Signer and hence it has no access to the needed secret. To keep the adversary as strong as possible, the adversary is given access to signing and sanitizing oracles, as well as a proof oracle. Hence, the adversary can control the input and observe the output of all secret based algorithms.

Note: For the obvious reason of modelling the property of transparency, the adversary's access to the proof-oracle (used for deciding which party is accountable) is limited, i.e., the use is excluded for the given (m, σ). *Brzuska et al.* [67] note that their previous definition in [64] did "[...] not consider the proof-restricted case. Without this restriction, though, achieving transparency at first seems to be impossible because the adversary can then always submit the replies of the Sanit/Sign oracle to the Proof oracle and thereby recover the secret bit b. However, in their construction the Proof algorithm searches in the list of previously signed messages and only gives a useful answer if it finds a match, enabling transparency without this restriction. Yet, any solution (like ours here) where the Proof algorithm is "history-free" can only achieve the proof-restricted version." [67].

The following formal definition is based on the proof-restricted one by *Brzuska et al.* as given in [67], the following version was jointly published with *C. Brzuska* and *K. Samelin* in [69] (see Appendix A publication n° 17).

```
**Experiment** $\mathsf{Transparency} - \mathsf{Brzuska} - \mathsf{et} - \mathsf{al.}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
    $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
    $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
    $b \leftarrow \{0,1\}$
    $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}}),\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit/Sign}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{sig}},\mathsf{sk}_{\mathrm{san}},\mathsf{pk}_{\mathrm{sig}},\mathsf{pk}_{\mathrm{san}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
        where oracle Sanit/Sign on input of $m_k, \mathsf{MOD}_k, \mathsf{ADM}_k$:
            if $\mathsf{MOD}_i \not\subseteq \mathsf{ADM}$, return $\perp$
            if $b = 0$:  first compute $\sigma_k \leftarrow \mathsf{Sign}(1^\lambda, m_k, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{ADM}_k)$,
                         then compute $(m'_k, \sigma'_k) \leftarrow \mathsf{Sanit}(1^\lambda, m_k, \mathsf{MOD}_k, \sigma_k, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$,
            if $b = 1$:  first compute $m'_k \leftarrow \mathsf{MOD}_k(m_k)$,
                         then compute $\sigma'_k \leftarrow \mathsf{Sign}(1^\lambda, m'_k, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{ADM}_k)$,
            finally return $(m'_k, \sigma'_k)$.
    return 1, if $a = b$ and $\mathcal{A}$ has not queried any $m_k$ output by Sanit/Sign to Proof,
    else return a random bit.
```

Figure 52. Proof-Restricted Transparency Experiment for SSS following *Brzuska et al.* [67]; as jointly published in *Brzuska, Pöhls, and Samelin* [69]

However, the ongoing work including joint works of the author of this thesis found that the above game in Fig. 52 does not completely cover the intended cases of the proof-restriction. Firstly, *Krenn et al.* [294] raise a stronger notion that relaxes the condition what an adversary can query to Proof to the message-signature pairs instead of just the message, i.e., instead of "$\mathcal{A}$ has not queried any m_k output by Sanit/Sign to Proof" [67] it becomes "$\mathcal{A}$ has [not] queried any (m', σ') output by Sanit/Sign to Proof" [107]. See the original works for an explanation of how this strengthening is helpful. Secondly, *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107] identified lately that the history-based Proof oracle's history can be controlled by the adversary and thus also a check of the input to the history must be included in the restrictions imposed on the Proof oracle. This results in the following definition, named strong transparency following *Krenn et al.* [294], which is also published in the latest full version[694] of the joint work by *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107] published at the PKC 2017 conference. It has been notationally adapted from the version in [107] to match the notation used in this thesis and to match the presentation in Fig. 52. The proof restriction that an adversary shall not break transparency with the help of the Signer and thus by using Proof on a message-signature pairs — following *Krenn et al.* [294] — related to the challenge of Sanit/Sign is now captured explicitly in the Proof′ oracle as follows: $((m,\sigma) \in \mathcal{Q} \vee \mathcal{Q} \cap \{(m_i,\sigma_i)\} \neq \emptyset$. So its neither the direct input to the Proof nor it is supplied as part of the history. With this restriction the adversary can no longer use the Proof as an aid to win. This captures the intended behaviour as already given in the previous works' high-level definitions of the transparency property. This thesis will use the notion of "transparency" to mean this version of a proof-restricted transparency.

```
Experiment $\mathsf{Transparency}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
    $(\mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
    $(\mathsf{sk}_{\mathrm{san}}, \mathsf{pk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
    $b \leftarrow \{0,1\}$
    $\mathcal{Q} \leftarrow \emptyset$
    $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}}),\mathsf{Proof}'(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit/Sign}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{sig}},\mathsf{sk}_{\mathrm{san}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
        where oracle $\mathsf{Proof}'$ on input of $m, \sigma, \{(m_i,\sigma_i) \mid i \in \mathbb{N}\}, \mathsf{pk}'_{\mathrm{san}}$:
            return $\perp$, if $\mathsf{pk}'_{\mathrm{san}} = \mathsf{pk}_{\mathrm{san}} \wedge$
                $((m,\sigma) \in \mathcal{Q} \vee \mathcal{Q} \cap \{(m_i,\sigma_i)\} \neq \emptyset)$
            return $\mathsf{Proof}(1^\lambda, \mathsf{sk}_{\mathrm{sig}}, m, \sigma, \{(m_i,\sigma_i)\}, \mathsf{pk}'_{\mathrm{san}})$
        where oracle $\mathsf{Sanit/Sign}$ on input of $m_k, \mathsf{MOD}_k, \mathsf{ADM}_k$:
            $\sigma \leftarrow \mathsf{Sign}(1^\lambda, m_k, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{ADM}_k)$
            $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_k, \mathsf{MOD}_k, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$
            if $b = 1$:
                $\sigma' \leftarrow \mathsf{Sign}(1^\lambda, m', \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{ADM}_k)$
            If $\sigma' \neq \perp$, set $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(m', \sigma')\}$
            return $(m', \sigma')$
    return 1, if $a = b$
    return 0
```

Figure 53. Proof-Restricted Transparency Experiment for SSS based on *Brzuska et al.* [67] with additions from *Krenn et al.* [294] and *Camenisch et al.* [107]; as jointly published in the full version by *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107]; notationally adjusted

In accordance with *Brzuska et al.* [64], this thesis's definition requires that $\mathsf{MOD}_i \subseteq \mathsf{ADM}_i$ is true. Hence, the experiments also need to check this for all queries to the above oracles as otherwise a trivial attack vector exists. In particular, if from $\mathsf{MOD}_i \nsubseteq \mathsf{ADM}_i$ Sanit correctly outputs $\perp$ then the adversary easily can distinguish this error from a "fresh" signature. Note, in earlier work by *Brzuska, Pöhls, and Samelin* [68] from 2012 this requirement is still missing (see Appendix A publication n° 11). This mistake has been rectified in later joint work in 2013 by *Brzuska, Pöhls, and Samelin* [69] (see Appendix A publication n° 17).

11.6.4.2. Transparency: Weak transparency by *Ateniese et al.* and invisible SSS

As discussed above in Sec. 11.6.4, the original definition of *Ateniese, Chou, de Medeiros, and Tsudik* [12] defines weak transparency that got formalised by *Brzuska et al.* in 2009 as the property of transparency given above as Definition 158. As well stated above, the thesis does not agree with some details of the definition of strong transparency as given by *Ateniese et al.*.[695] Despite that, this thesis agrees with *Ateniese et al.* "[...] that strong transparency is not always better." [12]. Especially, this thesis analysis yields that the current form of transparency in which subsequent authorized modifications are hidden from Verifiers will not increase the probative value of the MSS (see Shortcoming 3 in Sec. 12.3 and Sec. 17.3.3 for the results). From the point of the research question, the notion of

[694] The version of [107] updated 2017-12-13 contains this, see `https://eprint.iacr.org/2017/011/20171213:114927`.

[695] The discussion and criticism was part of the published joint work [68] from 2012 (see Appendix A publication n° 11).

transparency being strengthened is not increasing the probative value. As shown it even negatively impacts them, however they increase privacy. The Requirement 10 captures the desired cross implication between privacy and probative value: A mechanism MAY prohibit the identification of additional context information further to standard privacy but MUST not violate Requirement 4.

Nevertheless, it is of separate interest to have a property such that "[...] the adversary should not even be able to tell which parts of the message are potentially mutable." [64]. This has been introduced as potential future change detection (*FD) in Sec. 6.3.2. However note that *Ateniese et al.* [12] did not give a formal security proof for a scheme that provides their "strong transparency" [12][696]. They proposed that with stronger transparency one could build what they termed "invisible sanitizable signature" [12] and *Brzuska et al.* said it was "an overly strong requirement". A positive answer to that quest for an invisible sanitizable signature was given in 2017: *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* formally defined the property, gave a construction and proved it secure [107] (see Appendix A publication n^{o} 55).

11.6.4.3. Transparency: Strong transparency by *Krenn et al.*

The work of *Krenn et al.* from 2015 also strengthened the property of transparency. However, as discussed, already standard transparency would hinder the detection of subsequent modifications by Verifiers on their own and thus increase the difference between legally accepted CDSS and MSS. In line with the goal of this thesis to increase the probative value and argue for the equivalence of MSS to CDSS this thesis will trade cryptographic transparency for the property of non-interactive public accountability. Hence, the stronger transparency as defined by *Krenn et al.* [294] is not of further interest. This thesis would however like to note that their definition is in line with the one used in this thesis regarding scope and goal.

11.6.5. Unlinkability for SSS

Unlinkability was formally defined by *Brzuska, Fischlin, Lehmann, and Schroeder* in [67]. As a privacy extension *unlinkability* describes "the impossibility to use the signatures to identify sanitized message-signature pairs originating from the same source" [67]. Provided the adversary is "[...] given a signature for a sanitized message of two possible sources, the adversary cannot predict the actual original message better than by guessing [...]"[67] - "[...] even [...] if the adversary [...] provides the two source message-signature pairs and modifications of which one is sanitized [...]"[67]. Unlinkability is stronger than privacy, and as shown in [67]:

$$Unlinkability \Rightarrow Privacy\,.$$

It was also defined in the work of *Ahn, Boneh, Camenisch, Hohenberger, abhi shelat, and Waters* [6] as " [t]he inability to link derived signatures to their original sources [...]" [6]

Definition 214 : Unlinkability

> ***Unlinkability*** *makes it impossible for third parties to decide from two message-signature pairs if they come from one single source message-signature pair.*
>
> *Note: Information leakage through the message itself is out of scope.*
> *Note: Unlinkability is stronger than privacy.*

For example, assume the original message $m = A||B||C$ allows all three blocks to be admissible and two different valid sanitized versions are publicly available. If the SSS is **not unlinkable**, an adversary that sees $m' = A||d||e$ and $m'' = f||g||C$ and the resulting σ' and σ'' will have a chance to deduce from the adjusted signatures that they are coming from the same m. This attack is also given in [67]. EXAMPLE

Note, even if such a linking would be possible via the two signatures, the adversary could still not be able to produce a valid signature for a joint or merged message that contains more un-sanitized blocks then each of the two linkable message have, e.g., deriving a signature for $m' = A||d||C$ might still not be possible from seeing σ', σ'', m' and m''. Note, this thesis formalised this action as merging and explicitly adjusted the security model (see the MRSS framework in Sec. 14.2 on page 369 and the

[696] "Our construction only provides for weak transparency. Accordingly, we only provide a formal security model for weak transparency, in terms of an indistinguishability property." [12].

proposed mechanism *merge$\mathcal{RSS}$* in Sec. 14.11 on page 409). Note further, being able to detect the linkage or being able to merge blocks from potentially differently sanitized message that originated from one message-signature pair will not allow the adversary to produce a valid signature for a fresh merged message. So unlinkability is not a core security property of private and accountable SSS; see also the independence of unlinkability in Sec. 11.7.

Definition 159 : Unlinkability for SSS

> *An SSS is **unlinkable** if for any efficient algorithm $\mathcal{A}$ the probability that the following experiment Unlinkability$^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$ 54 returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ). [67]*

The adversary is only given access to the public keys. Hence, the adversary must use oracles for signing, sanitizing and generating proofs. The adversary is adaptive; it knows valid sets of $\{m_i, \mathsf{ADM}_i, \sigma_i\}$ from previous queries to the respective oracles.

The LoRSanit oracle works as follows: It uses the previously fixed signer key $\mathsf{pk}_{\mathrm{sig}}$ to generate from two such adversary supplied inputs two sanitized outputs. The game's underlying idea is "indistinguishability-based" [67]: From the output the two input messages shall not be distinguishable. In detail, the adversary provides the LoRSanit oracle with two inputs $\{m_0, \mathsf{MOD}_0, \mathsf{ADM}_0, \sigma_0\}$ and $\{m_1, \mathsf{MOD}_1, \mathsf{ADM}_1, \sigma_1\}$. Next, the oracle sanitizes a randomly chosen input by applying the supplied MOD. To avoid trivial attacks, the oracle ensure that not one choice of input produces an error and that either input would result in exactly the same output:

$$\mathsf{Sanit}(1^\lambda, m_0, \mathsf{MOD}_0, \sigma_0, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}}) = \mathsf{Sanit}(1^\lambda, m_1, \mathsf{MOD}_1, \sigma_1, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})\,.$$

```
**Experiment** $\mathsf{Unlinkability}^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{pk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
  $b \overset{\$}{\leftarrow} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}}),\mathsf{Proof}(1^\lambda,\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots)\mathsf{LoRSanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{sig}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
    where oracle LoRSanit on input of
    $m_0, \mathsf{MOD}_0, \mathsf{ADM}_0, \sigma_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}_1, \sigma_1$:
      if $\mathsf{ADM}_0 \neq \mathsf{ADM}_1$, return $\perp$
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}_0$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}_1$, return $\perp$
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{Verify}(1^\lambda, m_0, \sigma_0, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}) \neq \mathtt{true}$, return $\perp$
      if $\mathsf{Verify}(1^\lambda, m_1, \sigma_1, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}) \neq \mathtt{true}$, return $\perp$
      return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_b, \mathsf{MOD}_b, \sigma_b, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$
  return 1, if $a = b$
```

Figure 54. Unlinkability Experiment for SSS based on *Brzuska et al.* [67]

11.6.6. Accountability for SSS

Definition 107 : Accountability

> ***Accountability** allows a predefined set of entities to determine a valid signature's origin (Signer or Sanitizer) according to a protocol, i.e., decide which party is accountable (split in Signer-accountability and Sanitizer-accountability) for the signature of a given valid message-signature pair (m, σ).*

Following *Brzuska et al.* this thesis also splits accountability into Signer- and Sanitizer-Accountability [64]. The initial formalisation was given by *Brzuska et al.* [64]. Signer-accountability prohibits a sanitizer from being able to blame the Signer if indeed the Sanitizer is responsible for a given message. Vice versa, Sanitizer-accountability prohibits a Signer from being able to blame the Sanitizer if indeed the Signer is responsible for a given message.

Both, interactive non-public Signer-accountability and Sanitizer-accountability have been extended according to *Gong et al.*. This results in the description of the admissible parts (ADM) being explicitly stated inside the games [216].

Definition 160 : Interactive Non-Public Signer-Accountability (INT) for SSS

> *An SSS is* ***interactive non-public Signer-accountable****, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment Sig-Acc$_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 55 returns* 1 *is negligible (as a function of λ) and additionally the Judge algorithm requires a proof π generated by Proof needing sk_{sig} as input to the Proof.*

To break Signer-accountability the adversary needs to provide a proof π^* which makes a genuine Arbitrator decide that the Sanitizer is accountable, even though the Sanitizer is not accountable. In other words, breaking Signer-accountability means that an adversary can frame a Sanitizer as being deemed accountable by the Judge algorithm for a valid message-signature pair even if that Sanitizer has never acted upon this message-signature pair. The actions of a Sanitizer might change both, signature and message, or just one of them. The results shall be as follows:

- m and σ After changing the message's content in ways allowed by ADM only an authorized Sanitizer can also adjust the signature using the sanitization secret $\mathsf{sk}_{\mathrm{san}}$, such that the signature on the new message-signature pair stays valid.
- σ Generating only a valid signature involving $\mathsf{sk}_{\mathrm{san}}$ allows the Sanitizer to take accountability without a need to actually modification any of the message's content.
- m When only the message is modified this is equal to an unauthorized action that should result in an invalid signature.

When breaking Sanitizer-accountability the adversary's goal is as follows: maliciously accuse one Sanitizer that is identified and fixed by the key pair $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}})$. As the resulting signature must still verify to win, the adversary needs to change either only the signature alone or the signature-message pair. The adversary is given access to all secrets except the secrets of the party it needs to accuse of doing something that that party did not. Hence, the adversary is in the role of a Signer, with full control or access to $\mathsf{sk}_{\mathrm{sig}}$. In Fig. 55 the adversary has full control over the Signer's key pair, i.e., it is not generated by the experiment. However, the adversary has no access to $\mathsf{sk}_{\mathrm{san}}$, and hence is given access to an oracle sanitizing adversary supplied messages.

```
Experiment $\mathsf{Sig\text{-}Acc}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
  $(\mathsf{pk}^*, \pi^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sanit}(1^\lambda, \cdots, \mathsf{sk}_{\mathrm{san}})}(1^\lambda, \mathsf{pk}_{\mathrm{san}})$
    for $i = 1, 2, \ldots, q$ let $(\mathsf{pk}_{\mathrm{sig,i}}, m_i', \sigma_i', \mathsf{ADM}_i)$
    denote the answers and queries from and to the oracle Sanit
  return 1, if:
    $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}) = \texttt{true}$, and
    $\forall i \in \{1, \ldots, q\} : (\mathsf{pk}^*, m^*, \mathsf{ADM}^*) \neq (\mathsf{pk}_{\mathrm{sig,i}}, m_i', \mathsf{ADM}_i)$, and
    $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}, \pi^*) = \texttt{San}$
```

Figure 55. Interactive Non-Public Signer-Accountability Experiment for SSS based on *Brzuska et al.* [64] and *Gong et al.* [216]

Definition 161 : Interactive Non-Public Sanitizer-Accountability (INT) for SSS

> *A sanitizable signature scheme SSS is* ***interactive non-public sanitizer-accountable****, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment San-Acc$_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 56 returns* 1 *is negligible (as a function of λ) and additionally the Judge algorithm requires a proof π generated by Proof algorithm needing sk_{sig} as input to the Proof.*

In the game for interactive non-public Sanitizer-accountability from Fig. 56 the adversary has to generate a proof π^* which makes a genuine Judge algorithm decide that the Signer is accountable, even though the Signer is not accountable because the Sanitizer has acted on the message. Note, the execution of the algorithm Judge by the Arbitrator is the reason for this property being called interactive, because it requires interaction with the Signer, i.e., the Signer's participation due to the need for $\mathsf{sk}_{\mathrm{sig}}$ in Judge. In this interaction, a Signer failing or refusing to participate, i.e. provide some proof π that is invalid, will result in the Judge algorithm to accuse the Signer, as a party not obedient to the protocol. Hence, the adversary's attack would clearly manifest if the party carrying out the algorithm Judge would not also reliably request and obtain a genuine proof π from the genuine Signer. This thesis assumes that the Judge algorithm can be called and works correctly (see Sec. 11.4.3 for proof correctness involving the algorithms Judge and Proof).

```
Experiment $\mathsf{San\text{-}Acc}^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^\lambda)$
  $(\mathsf{pk}^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\cdots,\mathsf{sk}_{\mathsf{sig}},\cdots),\mathsf{Proof}(\mathsf{sk}_{\mathsf{sig}},\cdots)}(1^\lambda, \mathsf{pk}_{\mathsf{sig}})$
    for $i = 1, \ldots, q$ let $(m_i, \mathsf{ADM}_i, \mathsf{pk}_{\mathsf{san,i}})$ and $\sigma_i$
    denote the adaptive queries to the oracle Sign
  $\pi \leftarrow \mathsf{Proof}(1^\lambda, \mathsf{sk}_{\mathsf{sig}}, m^*, \sigma^*, \{(m_i, \sigma_i) \mid 0 < i \leq q\}, \mathsf{pk}^*)$
  return 1, if:
    $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathsf{sig}}, \mathsf{pk}^*) = \mathtt{true}$, and
    $\forall i \in \{1, \ldots, q\} : (\mathsf{pk}^*_{\mathsf{san}}, m^*, \mathsf{ADM}^*) \neq (\mathsf{pk}_{\mathsf{san,i}}, m_i, \mathsf{ADM}_i)$, and
    $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathsf{sig}}, \mathsf{pk}^*, \pi) = \mathtt{Sig}$
```

Figure 56. Interactive Non-Public Sanitizer-accountability Experiment for SSS based on *Brzuska et al.* [64] and *Gong et al.* [216]

11.7. SSS: Relations between security properties

For some properties relations have already been mentioned. *Brzuska et al.* [64] gives the relations of *Unforgeability, Immutability, Transparency, Accountability and Privacy* [64]. In particular, they proved that except for *Transparency* $\Rightarrow$ *Privacy* and the *Accountability* $\Rightarrow$ *Unforgeability* the properties are independent. Further, *Brzuska, Fischlin, Lehmann, and Schroeder* [67] extended the relations to cover also the property of *Unlinkability*, which is stronger than privacy, i.e., *Transparency* $\Rightarrow$ *Privacy*, but otherwise independent. *Brzuska et al.* [67] then added the property of *Unlinkability*, which is stronger than privacy, i.e., *Transparency* $\Rightarrow$ *Privacy*, the authors proved in [67]. The authors also prove that it is otherwise independent and does not affect the other properties.

The proofs of the relations for these state-of-the-art security properties are not restated in this section, but the interested reader is relegated to the original sources. This thesis presents several new properties or refinements of existing properties, a proof for these is provided in this thesis Chapters 13 and 14. Important in the context of the quest to offer an increased probative value is the need to achieve also the public form of accountability (PUB), and thus transparency can not be achieved. In this respect, it is especially interested that $\neg$ *Transparency* $\not\Rightarrow$ $\neg$ *Privacy*, which is stated as Theorem 2. Thus, carefully removing transparency while not negatively impacting privacy was the cryptographic challenge of the thesis, as briefly introduced in Sec. 3.10.2.3 on page 65.

Definition 162 : Transparency $\Rightarrow$ Privacy

> *"A transparent sanitizable signature scheme is also private." [64]*

Definition 163 : Signer- and Sanitizer-Accountability $\Rightarrow$ Unforgeability

> *"A sanitizable signature scheme which is sanitizer-accountable and signer-accountable is also unforgeable." [64]*

Note, *Brzuska et al.* [64] in particular state and prove that there are no more relations between the original five security properties. All the other properties from [64] are independent. Also, there are no relations between combinations of those properties. In the following independency declarations note that the authors call an SSS secure if "it is simultaneously immutable, unforgeable, private, transparent, sanitizer-accountable and signer-accountable" [64].

Definition 164 : Immutability is independent

> *"Assume that there exists a secure sanitizable signature scheme. Then there exists a sanitizable signature scheme which is unforgeable, private, transparent, sanitizer-accountable and signer-accountable, but not immutable." [64]*

Definition 165 : Transparency is independent

> *"Assume that there exists a secure sanitizable signature scheme. Then there exists a sanitizable signature scheme which is immutable, unforgeable, private, sanitizer-accountable and signer-accountable, but not transparent." [64]*

Definition 166 : Sanitizer-Accountability is independent

> *"Assume that there exists a secure sanitizable signature scheme. Then there exists a sanitizable signature scheme which is immutable, unforgeable, private, transparent and signer-accountable, but not sanitizer-accountable." [64]*

Definition 167 : Unlinkability is independent

> *"Assume that there exists a sanitizable signature scheme (obeying one or more of the properties unforgeability, immutability, privacy, (proof-restricted) transparency, signer-accountability and sanitizer-accountability). Then there exists a sanitizable signature scheme which is not unlinkable but preserves the other security properties." [67]*

Definition 168 : Unlinkability Implies Privacy

> *"Any unlinkable sanitizable signature scheme is also private." [67]*

In particular it is important for the further discussions in this thesis that:

Theorem 2 : $\neg$ Transparency $\not\Rightarrow$ $\neg$ Privacy

> *Assume that there exists a secure sanitizable signature scheme that is not transparent. Then there exists a sanitizable signature scheme which is immutable, unforgeable, sanitizer-accountable and signer-accountable, and private.*

Proof 2

> *This obviously follows from the independence of transparency given in Definition 165 and that transparency is stronger, but separated from privacy Definition 162. Both statements were stated and proven by Brzuska et al. [64].* □

11.8. RSS: State of the Art

A history of the terminology given in the literature was already presented in 3.4.2. In this section the works known to the author of this thesis in the field of RSS and closely related areas are presented[697]. They are sorted by year of publication. Works considered especially important or further links between them that are of extended interest have already been mentioned or will get mentioned in more detail when their contributions are discussed in more depth at different locations throughout this thesis. The state of the art in SSS is presented in Sec. 11.1. In Chapter 15 RSS get cryptographically separated from SSS.

In general, work considering homomorphic signature schemes is related to the field of RSS and SSS. *Boneh and Freeman* [57] present what is considered "[t]he first homomorphic signature that supports evaluation of multivariate, bounded-degree polynomials on authenticated data." [57]. This thesis sees functional signatures as different to RSS — and SSS. They offer to define a function and then allow to delegate the signing to third parties which can produce signature over all messages that fulfil the function. While it might be interesting to see if a functional signature could be used to emulate the functionality and cryptographic properties of RSS, the constructions come at higher costs than RSS, as they allow more general operations than redaction while giving more fine-grained control than just arbitrary sanitizations, e.g. multivariate, bounded-degree polynomials. This thesis will not list work from the pure field of functional signatures in this section. See [137, 475] for an overview of functional signatures.

Note, own or joint work involving the author of this thesis is not listed in this section; please see Appendix A for an overview of the author's works and Sec. 16.4 for a brief discussion that the work influenced the state of the art.

[697] This list contains work that was published before roughly the third quarter of 2017.

In 1989, *Merkle* [338] proposed an important concept also facilitated in RSS literature[698]. His work describes the verification of integrity of content by putting them as leaves in a tree where each parent node's content depends on the content of all its children. This is know as the Merkle hash tree (MHT) [338].

In 2002, *Steinfeld, Bull, and Zheng* [455] introduced the concept of removing parts from signed data without invalidating the signature was introduced as "content extraction signature" [455]. Their work contains a **formal** proof of security, including a formal description of the cryptographic property of privacy. The scheme works on ordered lists and allow "[...] the signer [...] to specify which extracted subdocuments the user is "allowed" to extract valid signatures for." [455]. See the discussion of their privacy definition in Sec. 9.1.2 on page 215. As discussed in Sec. 3.4.2, their work forms the basis for the principles of RSS used in this thesis, such as that redactions are public, the privacy definition, or the authorized modification policy "content extraction access structure CEAS" [455] denoted as ADM in this thesis (see Definition 48 on page 50). For privacy their proposed construction does not sign each block directly, but a commitment to the block (denote $m[i]$ in their work) concatenated with a random ($r[i]$). "To sign a document, one commits to the submessages and signs the concatenation of the commitments. The CEAS is appended to the signature and is also appended to each submessage before committing. The randomness strings used in the commitments are also appended." [455]. To achieve the privacy guarantee they "[...] require [the] CM [to] satisfy the standard hiding and binding security notions (we actually only need a relaxed binding requirement [...])." [455]. They formally specify all the required security properties, please consult the original work for details. The redaction consequently "[...] requires one to recompute the commitments to the removed (unextracted) submessages and append them. The randomness strings for the removed submessages are removed."' [455]. During verification the Verifier re-computes the commitments for the blocks still present and takes the supplied commitments for the redacted blocks. All of those were signed and the signatures including those over the policy (CEAS) are verified. Additionally, the conformance to the policy is checked. Their work — together with the work of *Johnson et al.* [273] described next — initiated the field of RSS research, and laid the basis for other works on security, e.g. [66], or generalisations and implementations of RSS, e.g. [444, 507].

Johnson, Molnar, Song, and Wagner [273] introduced the same concept as "homomorphic signatures" [273]. Their work appeared at the same time and independently of *Steinfeld, Bull, and Zheng* [455]. Their concept is to facilitate a Merkle hash tree MHT [338] to allow block encoded into the leaf-nodes to undergo verifiable redaction by the user. By signing the constant root-hash-value of the MHT they protect all blocks' content's integrity. By supplying the hash-value instead of the redacted content they allow the Verifier to re-create and compare the same root-hash-value based on the remaining non-redacted document's blocks and the supplied hash-values where necessary to reconstruct the hash-tree. See a discussion of their privacy definition in Sec. 9.1.3. Privacy is mainly achieved by not hashing the content of the block directly into the leaf-nodes of the MHT (denoted v in the work) but concatenate the actual content (x) with a random value (k). *Johnson et al.* [273] propose to use a GGM-tree[699] to generate those random values from a randomly chosen seed ($k_\in$). "The message is signed in three phases: first, we generate k-values by recursing down the tree with the PRG G; then, we generate v-values by recursing up the tree with the hash H; finally, we sign $v_\in$ using our conventional signature scheme." [273]. For a complete message — or a complete binary subtree — the Verifier is allowed to reconstruct the random values using the GGM-tree, i.e., the signature for a original message contains $k_\in$ and σ which was calculated by a secure signature scheme over the root of the MHT ($v_\in$). A redaction then removes the actual value x_i from the message and additionally removes the ability of the Verifier to calculate the corresponding random k_i by supplying the old random values for the remaining subtrees and the leaves where necessary. See the work for a detailed explanations and examples. The work contains a proof sketch, and "[...] the privacy security notion is not formalized [...]"[700] [457][701].

[698] Merkle trees appear for example in [141, 273, 461, 514].

[699] GGM is shorthand for the method proposed by *Goldreich, Goldwasser, and Micali* [209] that constructs a pseudo-random-function from a pseudo-random number generator which can be represented as a binary tree. Due to this representation it lends itself to construct random values that correspond to contents for the MHT which is also binary here.

[700] The American English spelling from original work is retained for terms and quotes.

[701] Actually, this note is found in [457], which is an updated version of [455] dated 2003.

In 2003, *Bull, Stanski, and Squire* [71] built an XML signature implementation for the content extraction signature of *Steinfeld, Bull, and Zheng* [455]. Interestingly they note, with respect to XML, that their "[...] approach to signing and verifying a document using dynamically loaded transforms further blurs the notion of what is being signed. We are not only signing the document content, but we are also signing functionality with respect to the content." [71].

In 2005, *Miyazaki, Iwamura, Matsumoto, Sasaki, Yoshiura, Tezuka, and Imai* [347] use the term 'sanitize' for the action of redaction. While their work's title is "Digitally Signed Document Sanitizing Scheme with Disclosure Condition Control" [347] the functionality of the authorized subsequent modifications described is what this thesis called a redaction (see Sec. 3.4.2), i.e., "masked" [347]. They add the functionality of being able to revoke only the right to redact from not yet redacted but redactable blocks. By this "disclosure condition control" [347] they allow to thwart an "[a]dditional sanitizing attack" [347]: With the redaction being a public operation an adversary can intercept a correctly redacted document. It could then maliciously additionally "delete[~~s~~] portions he/she deems undesirable, and forward[~~s~~] the additionally sanitized document to the requester." [347]. The Verifier is not able to detect that several redactions have been carried out, but it sees only the result after the sum of all redactions. A Sanitizer could revoke the Signer assigned authorization to redact a block, which would inhibit the malicious adversary to redact that block. The solution got termed 'consecutive redaction control' in this thesis and is discussed in Sec. 14.5.1.1 on page 379.

Izu, Kanaya, Takenaka, and Yoshioka [264] is titled "A Partially Sanitizable Signature Scheme" [264], while the terminology is different the described scheme allows only for a consecutive removal of blocks; thus [264] is in fact an RSS and not an SSS as the name might suggest. The authors clearly state their intentions regarding functionality as described by *Steinfeld et al.* [455][702]. However, the scheme is not private with a strength comparable to *Brzuska et al.* [64]. The authors state that their scheme achieves "secrecey [sic] (no leakage of sanitized information) [...] [s]ince a signer's signature *S* consists of a set of hash values and its signature, and a sanitizer's signature *S'* consisnts [sic] of an updated set of hash values and its signature, the secrecy of the sanitized information is assured by the preimage resistance of the hash function." [264]. The authors do not give a formal model for this property.
They also mention the solution termed in this thesis consecutive redaction control (see Sec. 14.5.1.1). as a solution to the "Additional Sanitization Attack" [264].

In 2006, *Miyazaki, Hanaoka, and Imai* [348] introduced a first security model for RSS. Their security model used set notions, which enables to facilitate mathematical notions from sets for explanations, like intersection and union. This can be done without the loss of generality. This thesis will also use set notation, as this notation allows defining message compositions as a union of sets and redactions as set intersections. This improves the readability (see for example Definition 35 in Sec. 3.2). Note that *Miyazaki et al.* [348] speak of "sanitize"[348], but the functionality described is what this thesis termed 'redact' based on different terminology as stated and motivated with other influential publications in Sec. 3.4.2. According to [348] the idea of "the document sanitization problem"[348] and "four digital document sanitizing schemes"[348] was apparently also published as tech. report in 2003 in Japanese [346][703]. This thesis follows a different naming; in general naming is not an issue, however, one problem with the difference in this naming is that it blurs that RSS and SSS are substantially different as discussed in Chapter 15.

In 2007, *Gentry, Ramzan, and Bruhn* [204]'s patent filed in 2005 on the use of redactable signatures in cryptographic digital certificates was published. It describes the use of an RSS to sign multiple parties into one certificate issued and signed by the certificate authority and then allow to redact them in order to not having to sent them to the Verifier. This usage is also aimed at an improvements of certificate revocation.

[702] *Izu et al.* [264] actually write "[t]he sanitizable signature scheme (or the content extractable signature scheme) [...]" [264] to explain their usage of terms; note that the work describing the sanitize functionality differently termed 'Sanitizable Signatures' by *Ateniese et al.* [12] is from the same year.

[703] The author of this thesis was unable to gain access to the original Japanese piece of work, nor to an English translation.

In 2008, *Haber, Hatano, Honda, Horne, Miyazaki, Sander, Tezoku, and Yao* [222] work is leaning more towards an RSS than an SSS, however it additionally allows for a limitation to a set of values rather than just redacting the content. The sanitization aspect was shortly described in Sec. 11.1 on page 248. As there might be the option for just substituting with a so-named "suppressed attribute" [222] this scheme is considered to be closer to an RSS. Their scheme features Signer control, i.e., which blocks can be redacted and which cannot is fixed during signing. They also mention the property of consecutive redaction control (see Sec. 14.5.1.1 on page 379). In their scheme the Verifier can see the position where the redaction has happened, this property is later detailed and defined in this thesis as weak privacy (see Sec. 15.1.4 on page 434).

Kundu and Bertino [295] presented a scheme to redact trees. As noted in *Brzuska et al.* [66] (see Sec. 12.1.4.1) there are attacks possible (see also Sec. 12.1.4.2) and the scheme "[...] does not achieve transparency [...]" [66] that it claims. Moreover, the scheme has a quadratic runtime complexity, that has been shown in an implementation presented in joint work [425] (see Appendix A publication n^{o} 10) to "[...] become unuseable [sic] very fast [...]" [425].

Miyazaki and Hanaoka [344], extending the work from 2005 [347] published by *Miyazaki, Iwamura, Matsumoto, Sasaki, Yoshiura, Tezuka, and Imai*, published their article with the title "Invisibly Sanitizable Digital Signature Scheme" [344]. The functionality of removing parts is termed 'redaction' in this thesis (see Sec. 3.4.2). This thesis categorises this scheme as an RSS and notes that the security models of RSS and SSS substantially differ as discussed in Chapter 15.

Izu, Kunihiro, Ohta, Sano, and Takenaka [265] proposed a scheme that can be identified as being weakly private, as defined in Definition 232 on page 434. This work was published in proceedings in 2009. Following the explanation of the functionality offered by the schemes, the authors works get classified under RSS, while the title reads "Sanitizable and Deletable Signatures". The authors describe — without formalising those properties or relating to the existing formalisation *Steinfeld, Bull, and Zheng* [455] — that their schemes allow that "[...] the secrecy of sanitized parts is assured, namely, no information of sanitized parts will be leaked after the sanitizations."[265]. This matches the high-level description of the goal behind the cryptographic property called privacy in this thesis. The authors note that still "[...] a verifier can detect whether the document is sanitized or not, moreover, which subdocuments are sanitized." [265]. They term the missing operation "delete" [265] and quote *Miyazaki et al.* [348].[704] This thesis terms the cryptographic property that a deletion would achieve transparency following [66] and others[705] defined in Sec. 11.13.3 on page 288. Indeed, *Izu et al.* already introduce this as a notion that is stronger than the property they termed it "secrecy" — comparable to what this thesis calls cryptographic privacy following *Slamanig and Rass* [441], *Canard et al.* [112], *Attrapadung et al.* [15] , or *Kundu and Bertino* [299]. Their schemes would likely not achieve the transparency notion as defined in Sec. 11.13.4 on page 290 as an attack — similar to those *Brzuska et al.* [66] had described to break *Kundu and Bertino* [296] (see Sec. 12.1.4.1) — based on the increasingly ordered random ID's would allow statistical attacks to identify the gaps where likely blocks have been deleted. However, as the deletion would remove any hashes that are based on the original content of the blocks it upholds privacy; only the fact that redactions have taken place might be not statistically hidden. This property is called weak privacy in this thesis and introduced in Chapter 15 in Sec. 15.1.4 on page 434.

In 2009, *Bauer, Blough, and Mohan* [28] extend the work of *Johnson et al.* [273] from 2002 by including trees containing data from multiple authorities in which each authority signs a subtree of its data.[706] They note that their work is focussed on releasing only a small amount of data by redacting the majority [28]. As in the original work their redaction capabilities are based on the MHT's root's content or the content of nodes with low depths are not changing and thus signing them with a standard CDSS offers integrity and authenticity protection for all leaf-nodes. Additionally, they cater for referential integrity protection (see Definition 116 on page 166), as their scheme offers that redactable signature is able to "[...] prevent [verifiable signed] data from being released without respecting the relationships between

[704] Calling it delete is not a misnomer per se, but in their description they state "[t]he deletable signature [...] is a digital signature scheme in which a subdocument (and corresponding data) can be deleted without keeping [sic] the integrity of the remaining subdocuments. " [265]. The opposite is the desired goal: To allow deletion while keeping the integrity of the remaining subdocuments.

[705] Cmp. [508].

[706] A year before *Bauer, Blough, and Cash* [27] with reference to the work of *Steinfeld et al.* [455] build an anonymous credential system based on redactions. However, that work is not on a redactable signature scheme per se.

different pieces of data." [28]. The Signer can define at signature generation that certain blocks' contents must either be released together or not at all, as "[...] releasing individual pieces of data ignorant of those relationships [...] makes no sense." [28]. If not released together as specified the signature would not verify.

Chang, Lim, and Xu [117] presented a scheme for trees that allows to protect the structural integrity of the tree's siblings, i.e., left-to-right order between the child elements of a node. They preserve the order the ordering between siblings is explicitly signed using a 'left-of' relation. They further strengthen the privacy notion into what this thesis denotes as *transparency*. To achieve this they require $\mathcal{O}(n^2)$ operations. This shows that transparent schemes are costly[707].

Slamanig and Stingl [444] present their scheme's practicality with a concrete use-case of health related documents. Namely, they discuss a redactable scheme for XML structures conforming to the clinical document architecture (CDA) and thus provide a very practical motivation for RSS applications. Their scheme is as such for tree-based data and is a generalisation of the one given by *Johnson et al.* [273]. They transform an arbitrary XML document — a tree structure that is in general neither binary nor complete — uniquely into an N-ary tree. The authors present a construction that generalises the Merkle hash tree as well as the GGM-tree used in the original work by *Johnson et al.* [273]. A proof of security of the construction or the properties is not provided in [444].

Kundu and Bertino [296] presented a redactable signature scheme for trees. They motivate that "it is crucial that integrity and confidentiality be assured not only for the content, but also for the structure." *Kundu and Bertino* [296]. However, their scheme achieves only the proposed notion of weak privacy (see Sec. 15.1.4 on page 434). It has several weaknesses, in 2010 the analysis of several schemes under a newly defined privacy notions by *Brzuska et al.* [66] identified that the scheme by *Kundu* and *Bertino* [296] lacks provably transparency since it uses an order-preserving transformation of traversal numbers to random numbers [66] (see Sec. 12.1.4.1). Another attack on the privacy of the scheme has been proposed in this thesis in Sec. 12.1.4.2 and published in 2012[708].

In 2010, *Brzuska, Busch, Dagdelen, Fischlin, Franz, Katzenbeisser, Manulis, Onete, Peter, Poettering, and Schröder* [66] published a formalisation, comparable to theirs for SSS [64], for redaction over tree-based data structures. The work defined and formalised a set of desired properties for redactable tree-structured documents [66]. Note, both models offering about the same strength was among the reasons for selecting their formalisation as a basis for the harmonised definitions. They strengthened the privacy notion into what they called *transparency*. They "show that transparency is strictly stronger than privacy", i.e, like in SSS also for RSS Transparency ⇒ Privacy. But, importantly to disable transparency as suggested in Requirement 4[709], *Brzuska et al.* [66] show that privacy can be upheld in the absence of transparency. To achieve transparency they require $\mathcal{O}(n^2)$ operations, hence they show that transparency enhances privacy but that one needs to endure additional costs to achieve it. Notably, *Brzuska et al.* [66] also showed that most of the schemes proposed till then were not privacy preserving according to their model of privacy in [66]. A brief overview of schemes not private — and hence incapable to fulfil Requirement 9[710] — is provided in Tab. 9 in Sec. 12.1 on page 294.

Slamanig and Rass [441] published their work with the title "Generalizations and extensions of redactable signatures with applications to electronic healthcare" [441]. Their proposed scheme is based on the scheme given by *Johnson et al.* [273], which was for binary MHTs in order to secure linear documents. *Slamanig and Rass* [441] generalise this scheme to work with documents with a tree as the underlying data structure. Hence, they handle each node from the electronic health record, which is represented in XML, as a node within a generalised N-ary MHT and use a generalised GGM-tree for the generation of the random numbers, analogous to [273]. This generalisation is shown to preserve the security of *Johnson et al.* [273]'s scheme as a supplied sketch of a proof explains.

[707] Note, [117] itself does not offer a performance analysis on real data For another scheme with quadratic runtime complexity an implementation shows that those schemes can "[...] become unuseable [sic] very fast [...]" [425]; this is presented in Sec. 14.6.3 and was published in joint work [425] (see Appendix A publication n° 10).

[708] See [425] (see Appendix A publication n° 10).

[709] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[710] Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength) (see Sec. 10.1).

Kundu and Bertino [297] published a scheme that extends their previous work on tree-structured data [295, 296] to arbitrary graphs. In their work they enable the redaction of sub-graphs from signed graphs. It is based again on "[...] a randomized notion of such variants of pre-order numbers [...]"[711] [297]. Which might lead to the same problems as the randomised traversal numbers that were shown for [296] in the same year by *Brzuska et al.* [66].

Wu, Hsueh, Tsai, Lai, Lee, and Chung [507] provide a web based approach to medical document redaction. They provide the ability to redact through a concatenation of every redactable block of the document opposed to the Merkle hash tree (MHT) approach. Unfortunately, they neither provide details of the granularity provided by their method nor computation times required by their system[712].

In 2011, *Izu, Izumi, Kunihiro, and Ohta* [266] proposed a scheme that is only weakly private (see Definition 232). Following the explanation of the functionality offered this is an RSS that is private and transparent — different to the title that reads "Yet Another Sanitizable and Deletable Signatures". It is an extension of previous work by *Izu et al.* [265] from 2008. Again, the authors describe the functionality as follows: "In addition, the secrecy of deleted subdocuments is established. However, different from the sanitizable signature, a verifier cannot detect whether subdocuments were deleted or not." [266]. This thesis terms the cryptographic property that a deletion would achieve transparency following [66] and others[713], indeed *Izu et al.* already introduce transparency as a notion that is stronger than a property they termed "secrecy" — comparable to what this thesis calls cryptographic privacy following *Slamanig and Rass* [441], *Canard et al.* [112], *Attrapadung et al.* [15] , or *Kundu and Bertino* [299].

Lim, Lee, and Park [314] present a more efficient constructions, i.e. with a shorter signature, compared to a trivial construction that signs each blocks with an extra signature generation key. The authors share the thesis' high-level understanding of privacy: "The privacy requirement of redactable signatures is to guarantee the confidentiality of the content of erased message blocks from a signature of the redacted document." [314]. However, they define and prove their optimised schemes only secure under what they define as 'weak privacy'. This is defined different to a definition of this thesis (see Sec. 15.1.5 on page 435). The weak privacy of *Lim et al.* [314] prohibits adversary access to a signing oracle. As a result an adversary against a real deployment of the scheme must be prohibited from obtaining further signatures on messages of his own choosing, i.e. no chosen message attacks on privacy. This thesis aims to create MSS that can be facilitated to remove information from authentic data in a form that is legally accepted as a data protection enhancement. Thus, when looking for a cryptographically sufficient privacy level of the MSS the exclusion of this attack vector by *Lim et al.* [314]'s definition is problematic for the following reasons: First, because other notions like unforgeability for standard signatures, e.g., UNF-CMA[714] and other RSSs' definitions of privacy do not have this limitation. Second, in real-world deployments an attacker might — by error or by design — become able to trigger the generation of a signature for adversary-influenced inputs[715].

Miyazaki and Hatano filed a US patent back in 2006, which was published in 2011, on a system implementing RSS like behaviour giving a specific construction [345].

Ahn, Boneh, Camenisch, Hohenberger, abhi shelat, and Waters [5] generalised the concept of redaction inherent to RSS and presented schemes with the property later termed "context-hiding" appeared. Context-hiding is a very strong privacy notion, extending the accepted notions of transparency and privacy given for example by *Brzuska et al.* [66]. In this notion a derived signature does not leak whether

[711] The American English spelling from original work is retained for terms and quotes.

[712] Note, in the ReSCUeIT research project (2010–2013) a web-service-based implementation of an SSS for XML was implemented; see Sec. 18.1 for a description and actual runtimes.

[713] Cmp. [508].

[714] See Sec. 5.3.4 on page 125 for a description and definitions of existential UNF-CMA or EUF-CMA from *Goldwasser, Micali, and Rivest*.

[715] Compare the transformation of "[...] a [...] Web Service into a "Bleichenbacher oracle" [...]" [268] in the domain of encryption as an example.

it corresponds to an already existing signature in a statistical sense. This statistically hides authorized subsequent modifications from the Verifier, thus does not achieve to even detect the bit that a subsequent change has happened and thus fails Requirement 4[716]. Note, their work later appeared also as [5–7].

In 2012, *Lim, Lee, and Park* [314] proposed a scheme that is only weakly private (see Definition 232) as the work on separation Chapter 15 will show.

Attrapadung, Libert, and Peters [15] continued to build context-hiding schemes as introduced in [5]. Their scheme only allows for quoting instead of arbitrary redactions, i.e., redactions are only allowed at the beginning, or end resp., of a list of blocks[717].

Ahn, Boneh, Camenisch, Hohenberger, abhi shelat, and Waters [6] published parts of the work that was listed previously with reference to the preliminary published full version [5] from 2011 in this work [6]. The authors discuss their strengthening of the privacy property by their newly proposed stronger notion of statistical unlinkability. This still keeps occurred authorized subsequent modifications hidden from the Verifier, thus does not achieve to even detect the one bit of information that a subsequent change has happened and thus fails the legally induced Requirement 4[716].

Attrapadung, Libert, and Peters [15] further strengthen *Ahn et al.* [6]'s definition of unlinkability and achieve "[...] adaptive context hiding and complete context hiding, basically saying that derived signatures (for messages with any valid signatures) and fresh signatures are close." as *Deiseroth et al.* [136] have stated it. In a later paper the authors explained the strengthening as follows: "The difference between the definition of Ahn et al. [6] and the one of [*Attrapadung, Libert, and Peters*] [15] lies in that the former requires the unlinkability of derived signatures to only honestly generated signatures. In contrast, the stronger complete context hiding property [15] requires unlinkability with respect to any valid signatures, including those signatures that might have been somehow maliciously re-randomized by the adversary."[718] [16].

Finally note, due to the strong privacy and the cryptographic transparency a Verifier is not able to judge, also not with the help of the Signer to identify authorized subsequent modifications. While this is a strong privacy guarantee, such a scheme cannot fulfil Requirement 4[716]. The Signer will stay accountable unless an unauthorized modification happens.

Izu, Takenaka, Yajima, and Yoshioka [267], like in [264], called their work's functionality "sanitized signatures" [267] while it only allows to remove blocks. The functionality of removing parts is termed 'redaction' in this thesis (see Sec. 3.4.2). This difference in naming is especially important to uphold as the concept of "sanitizable signature schemes" coined in 2005 by *Ateniese et al.* in [12] is different from that of [267], which is an RSS. The aims of RSS and SSS seem to differ only slightly, but their security models substantially differ as discussed in Chapter 15.

Kundu, Atallah, and Bertino [300] introduced a redactable scheme for tree-structured data with a worst-case complexity of $\mathcal{O}(n^2)$. It has limited flexibility as it only allows to remove leaf-nodes and therefore offers **no** advantage to the one introduced by *Brzuska et al.* [66] in 2010. Note, in contrast to other schemes in previous work by the same authors [295, 296] the work [300] is built on a different idea, as it is based on signing binary relations, similar to [66, 117].

Unclear naming of works might have added confusion and might have hindered adoption in the past. Initial early works might have not termed it "redactable" [455] following *Steinfeld et al.* in 2002. However, the terminology could have stabilised in MSS after the term "sanitizable signature schemes" was used in 2005 by *Ateniese et al.* [12] or at latest after both got formalised by *Brzuska et al.* [64] in 2009 and *Brzuska et al.* [66] in 2010. However, the functionality that this thesis calls redactable following *Steinfeld et al.* [455] (see Sec. 3.4.2) was termed 'sanitizable' most recently in 2012 in [267] from 2012 following in line of earlier works [264, 344, 347, 348].

[716] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[717] For example, from the ordered list containing three strings [I, like, fish] the following ordered lists are considered valid quotes: [I, like, fish], [I, like], [like, fish], [I], [like], [fish] and ∅.

[718] Citation endnotes adjusted.

In 2013, *Attrapadung, Libert, and Peters* [16] designed a schemes that only allows quoting[719] The authors followed their line of work from 2012 [15] and kept a very strong statistical transparency and a statistical privacy property for RSS. The schemes in [16] do not offer non-interactive public accountability and they fail Requirement 4[720] which is proposed as required for an increased probative value. Their scheme again allows to quote substrings, i.e., the message is a linear ordered list of blocks and a valid redaction must retain a list containing only consecutively indexed blocks, e.g., quoting from a signed $m = a|b|c|d$ would allow $a|b$ or $b|c|d$ to still verify, but not $a|c$ or $b|d$.

Deiseroth, Fehr, Fischlin, Maasz, Reimers, and Stein [136] published work that leans towards functional signatures schemes as their authorized modifications are defined as P-homomorphic signatures with adjustable predicates. Their work extends work by *Ahn et al.* [6] and *Attrapadung et al.* [15] from 2012. It is listed here, firstly because their work shows how to build predicates over combinations of several P-homomorphic signatures each of the combined schemes could describe a redactable signatures as given in [6, 15]. The authors describe the "[...] possibility to combine a set of given homomorphic schemes for predicates $P_1, P_2, ...$ into one for a new P-homomorphic signature scheme. Here, P may be a simple combination such as $P_1 \vee P_2$ or $P_1 \wedge P_2$, or describe even more complex functions. An example are two redactable schemes, one allowing for redaction only at the front of the message (P_1), and the other one enabling redaction only at the end (P_2). Then a $P_1 or P_2$-homomorphic scheme would be a scheme for quoting substrings, by first pruning at the front and then truncating in another step at the end." [136]. This might be of separate interest as it shows a way in which one can construct redactable signature schemes with more complex policies. Secondly, because the research shows that recent research continued to keep the stronger privacy notion [6, 15]. Because of the statistically close distribution of valid derived signatures for modified messages and fresh' signatures for the original message allows to derive a strong privacy notion. "This guarantees for instance that the original message in case of redactable signatures remains hidden." [136]. The privacy guarantees are fulfilling those formalised by *Brzuska et al.* [66] in 2010 and the results of this thesis regarding the confidentiality protection apply. However, with the stronger privacy of strong context hiding schemes from this strand of research do not allow to build an interactive, let alone a public detection of subsequent authorized modifications, which makes it impossible to fulfil Requirement 4[720] required for an increased probative value.

Kundu and Bertino [299] presented a slightly modified scheme, compared to their previous works [295, 296, 300], for data that is structured as a graph. However, the scheme presented in [299] is considered not to fulfil the privacy notion formalised by *Brzuska et al.* [66]. Further the authors state in [299] that using accumulators is not sufficient, however the instantiation of $merge\mathcal{RSS}$ proves the contrary: Cryptographic accumulators are indeed sufficient to instantiate a secure RSS.

In 2014, *Stuart Haber* [461] introduced a "[...] quantified version of the transparency property, precisely describing the uncertainty about the number of redacted subdocuments that is guaranteed by the [...] scheme[~~s~~] [...]" [461]. This is an interesting observation and it is in line with the thesis finding that negating the property of transparency without impacting on its related property of security in a construction requires care. This challenge was presented briefly in Sec. 3.10.2.3 on page 65. The authors describe their scheme's operation as follows: "[...] we sign a sequence of randomized commitments that depend on the contents of the subdocuments of the document to be signed. In order to hide their number and location, we randomize their order, and mix them with a sequence of "dummy nodes" that are indistinguishable from commitment values. Our first scheme uses a data structure of size quadratic in the number of subdocuments, encoding all the precedence relations between pairs of subdocuments."[721] [461]

Backes, Dagdelen, Fischlin, Gajek, Meiser, and Schröder [19] offer a generalisation termed 'operational signatures schemes' which is general enough to encompass many existing schemes including RSS as well as SSS. The work gives concrete explanations how the general notation can be used to describe to RSS and SSS. However, [19] does not give the actual predicates for redaction and sanitization. Their

[719] Quoting means that redactions are only allowed at the beginning and/or at the end of an ordered data structure. See Footnote 717 for another set of example quotes from an sequentially ordered list.

[720] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[721] The American English spelling from original work is retained for terms and quotes.

privacy notion is based on the indistinguishability idea, but got generalised for predicates. In a nutshell their game lets the adversary supply two messages m^1 and m^2 and respective predicates P_0 and P_1 which after applying the predicates to the respective message must result in the same output. The authors further distinguish between a version for symmetric algorithms generating MAC instead of digital signatures and define a weaker notion which in [19] means that the inputs must both be validly signed. They state that their privacy notion "[...] implies several privacy notions in the context of redactable signature, for example [*Brzuska et al.*] [66][722]." [19]. In their notation the difference between the predicate for sanitization p_{san} and the one for redaction p_{red} is that the redaction requires no key and "[g]iven a message and a valid signature over the message, any party may delete parts of the message and infer a valid signature." [19]. While the predicates look quite similar in their notation, the authors' statement that "Redactable signatures [CLX09, BBD+10, SR10, SPB+12] are special sanitizable signatures with "public sanitization"." could mislead the unfamiliar reader: It is of course also a difference in the functionality of how the predicate changes m_{in} into m_{eval}: SSS allow to substitute while RSS allow only to remove.

Arshad, Kundu, Bertino, Madhavan, and Ghafoor [10] published work on the secure redaction of graphs, including two privacy notions. Those were recently shown to be too weak for practical application by *Erwig et al.* [165] in 2017. Thus, this scheme is no longer considered as it can be argued that it might not be secure enough to meet legal standards due to the weakness shown by the attacks from *Erwig et al.* [165]. Thus it could fail Requirement 7.2[723].

Slamanig, Stranacher, and Zwattendorfer [445] explain in their work how redactable signatures schemes and 'blank signatures' as introduced in [224] can be used to enhance the privacy of national electronic identity systems. "Many qualified eIDs issued and rolled-out by various countries miss the feature of selective disclosure in current implementations. This means that users usually have to reveal their full identity during an identification and authentication process even if a service provider only needs a subset of the identity for providing a service." [445]. Their approach uses several other cryptographic primitives, but especially redactable signatures are integrated to hand "[...] users [...] full control of their data, i.e., providing selective disclosure [...]" [445]. *Slamanig et al.* [445] show that exchanging standard with malleable signatures in-place allows to enhance an existing eID service without having to re-design the service-architecture: "[...] by using a malleable signature scheme, we can guarantee that only required (encrypted) attributes can be disclosed to the service provider without invalidating the signature of the trusted registration authority. Finally, the requirement of easy integration into existing infrastructures can be met by this model as existing identity protocols already support [...] digital signatures out-of-the-box." [445].

In 2015, *Idalino, Moura, Custódio, and Panario* [238] described partial integrity as the possibility to detect and locate modifications and highlight the benefits for several applications, but their scheme fails to achieve the notion of privacy for RSS given by *Brzuska et al.* [66] (see Sec. 12.1.1).

Haber, Horne, and Zhang [223] filed a patent in 2014 on a system for redactable document signatures based upon commitments for each blocks and related to the work published as [222]. The patent describes the generation of blocks and their commitment values, e.g. hashes with random values generated by a "[...] random value engine [that] can use a GGM-tree [...]" [223]. The commitments are put on an ordered graph, i.e. a "binary hash tree" [223], and "[...] the root of the binary hash tree can be submitted to a signature process of a digital signature scheme, and the output of the signature process can be the signature value [...]" [223].

Camenisch, Dubovitskaya, Haralambiev, and Kohlweiss [105] introduced an unlinkable RSS which is secure in the framework of universally composable security (UC) [113, 373]. They do so in order to build an anonymous credential system on it. The work was also published as [106]. With their proposed scheme "[...] one can redact message-signature pairs and reveal only their relevant parts each time they are used. Moreover, signatures in URS are unlinkable and the same message-signature pair can be redacted and revealed multiple times without being linked back to its origin." [105]. This thesis notes that the specific usage of the RSS for anonymous credentials is out of scope, as is designing schemes that satisfy UC.

[722] Original's endnote has been adjusted to correctly point to the same entry found in the bibliography of this thesis.

[723] Requirement 7.2 (Mechanism MUST be as good as the legally accepted current state of practice for integrity and authenticity protection) (see Sec. 7.7.2)

Krenn, Samelin, and Sommer [294] detailed many of the definitions making them more precise. Especially for the privacy definition, this thesis added the very sensible addition of additional checks of admissibility of both of the adversary's requested modifications from *Krenn et al.* [294]. They note that in the original work of *Brzuska et al.* [64] the messages sent by the adversary are never explicitly checked against the admissible change policy that is given jointly for both messages. Namely, the original "[...] only requires that MOD_0 and MOD_1 are compatible with ADM, while "the resulting modified messages are identical for both tuples" [64]. However, it is not clear what this concretely means if ADM does not match one of the messages."[724] [294]. Under the assumption of the worst behaviour of the algorithms in [64] — which were identified as underspecified by *Brzuska et al.* [64] — an adversary can craft a message pair where it can distinguish which input is used by the oracle in the privacy experiment. *Krenn et al.* [294] do assume that this was not intended stating that they "[...] assume that prior art, e.g., [64, 67], implicitly requires that $\mathsf{MOD}(m)$ returns $\perp$ if $\mathsf{ADM}(\mathsf{MOD}) = 0$. In our definition, we made this explicit."[724] [294]. This thesis added this to the standard privacy, to make this assumed and useful behaviour of *Brzuska et al.* [64] explicit. In the case of strong non-interactive public accountability the authors note: "[...] our strong definition also takes the signature into account." [294]. Again, this is a useful strengthening as it is especially helpful "[...] in the context of re-randomizable signatures or similar primitives [...]" [294]. And it can be achieved by a proposed scheme, namely $pubacc\mathcal{SSS}$ achieves *Krenn et al.* [294]'s " non-interactive public accountability." [294][725]. They also provide an even stronger transparency notion. However, this strengthening — like already the standard transparency — is negatively affecting the ability of the $\mathsf{Verifier}$ to detect authorized subsequent modifications and thus negatively affects the legal value as it does not offer non-interactive public accountability (see Sec. 7.3.2 and related discussions in Requirement 4[726]).

Ahn, Boneh, Camenisch, Hohenberger, abhi shelat, and Waters [7] published a journal article in line with their work that was en-listed already earlier as [5] in 2011 [5–7].

In 2016, *Ateniese, Magri, Venturi, and Andrade* [13] present the useful applications of the redaction functionality offered by RSS in general to the blockchain. Their approach is using a chameleon hash design, which is also generalised in their paper. The paper shows the general usefulness and manifold applications of redacting authentic content. It further mentions content that is illegal or no longer needed to hold due to the GDPR as reasons why content stored maliciously or due to historic transactions openly in the blockchain could be redacted from the public record (the blockchain). It got also published as [14] in 2017, but it does not discuss cryptographic requirements based on a deep legal analysis. However, in order to allow a minimum of legal transparency any redaction of the blockchain "[...] leaves an immutable "scar" to indicate when any blocks have been altered [...]" [14]. This can be seen as equal to the EU regulations requirement to an cryptographic integrity mechanism to enable the detection of any subsequent modification. The way this is done is described in a whitepaper as follows: "To positively identify blocks that have been changed, it is possible to architect blockchains so that any redaction leaves an inevitable "scar" that cannot be removed [...] by including both an editable chameleon hash to connect the blocks alongside a standard, uneditable hash." [323]. While the details are not analysed in this thesis, the way of implementing block-level[727] accountability to publicly identify modified blocks (described in [323]) it seems that the original content's hash is still present in the signature after a redaction. For RSS this is a potential data leak and prohibits the scheme to reach a level of standard privacy as defined by *Brzuska et al.* [66]. The impact of this information leak of redacted content through its remaining standard hash needs to be analysed in the chosen application of the redactable blockchain. This was not in the scope of this thesis. However, note that a privacy notion like standard privacy (as defined in Definition 174) based on an indistinguishability experiment with an adaptive adversary in the standard model is hard to reach, but offers nice cryptographic privacy.

[724] Original's endnote has been adjusted to correctly point to the same entry found in the bibliography of this thesis.

[725] See the discussion in Sec. 13.1.2for further details.

[726] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[727] Here meaning the blocks of the blockchain, but in this application they are equal with the notion of block from the realm of RSS.

In 2017, *Ma, Liu, Wang, and Wu* [325] published their work on An Efficient and Secure Design of Redactable Signature Scheme with Redaction Condition Control'. Note, they already cite and incorporate notions and results from this thesis[728]. Their "Redaction Condition Control" allow the Signer to initially specify which blocks are ""Redactable" (RED for short) and "Prohibited to be removed" (PRE for short)" [325]. Then subsequently each redacting Sanitizer could either remove a block that is marked as redactable (denoted as " "Removed" (REM for short)" [325]) or decide to prohibit its consecutive removal by a subsequent Sanitizer, i.e., it allows the following transitions: RED → REM, RED → PRE. This property is discussed under the term of 'consecutive redaction control' in Sec. 14.5.1.1. *Ma et al.* [325]'s notion of privacy matches that of *Brzuska et al.* [66] used in this thesis. Their scheme offers "[...] privacy and unforgeability of the signed messages and the transparency of redactable signature [...]" [325]. "This scheme has the security properties of unforgeability, privacy, and transparency, which are formally defined and proved. Compared with state-of-the-art redactable signature schemes, our scheme is more efficient in communication and computation cost." [325]. Note, this work already accepted and thus incorporated published results from this thesis.[729]

Erwig, Fischlin, Hald, Hehm, Kiel, Kübler, Kümmerlin, Laenge, and Rohrbach [165] further researched secure RSS for graphs. This extends the applicability of RSS to the more complex data structure of a graph. They identify an insecurity of the schemes by *Arshad et al.* [10] presented in 2014 and show by example "[...] that the two privacy requirements [from [10]] are too weak." [165]. *Erwig et al.* [165] also acknowledge that flexibility of the redaction capabilities towards structure itself are important and thus their scheme allows for a "[...] redaction [that] is quite general in the sense that one can also, say, redact edges for a graph." [165]. This is in line with the analysis of shortcomings presented in this thesis as Shortcoming 5[730] regarding the limited flexibility to controllably allow subsequent modifications to the structure only. This thesis proposed $struct\mathcal{RSS}$ (see Sec. 14.9.1) which was jointly published in 2012 [426] as a solution towards making structure, i.e. edges, a first class citizen for redaction in an RSS for tree-based data structures. This exact work, i.e., [426], — and also [425] — got acknowledged in *Erwig et al.* [165]'s related work.

11.9. RSS: Harmonised high-level description

Following the related work, this thesis sees redactable signature schemes (RSS) in the same way as described by *Johnson et al.* in [273]:

> "Redactable signatures are intended to model a situation where a censor can delete certain substrings of a signed document without destroying the ability of the recipient to verify the integrity of the resulting (redacted) document. In particular, we allow the censor to replace arbitrary bit positions in the document with a special symbol # representing the location of the deletions. (Our construction can be readily generalized to any alphabet, so that the signer can limit redactions to whole words, sentences, etc., but for simplicity we describe only the case of bitstrings here.)"[731] [273]

Henceforth, an RSS allows a third party to remove blocks of a signed message m without invalidating its protecting signature σ. This action can be performed without involvement of the Signer and it needs no additional secrets, i.e., redaction is a public operation. The signature on the redacted message m' still verifies under the Signer's public key. Assume, for the sake of readability but without loss of generality, that the message is represented as set of blocks, i.e., $\mathcal{S} = \{m[1], m[2], \ldots, m[\ell]\}$. In this setting an RSS would allow any third party to remove elements from the signed set, e.g., $\mathcal{S}' = \mathcal{S} \setminus m[i]$, and adjust the signature to σ', such that $(\mathcal{S}', \sigma')$ verifies under $\mathsf{pk}_{\mathsf{sig}}$, as long as the Signer authorized this change from $\mathcal{S}$ to $\mathcal{S}'$. As a result, the Verifier only sees a blinded version of the message, while still being able to verify that the remaining blocks, those that the Verifier was supplied with, are still unchanged and authentic. The workflow was shown in Fig. 18.

In RSS redaction is a public operation. In an RSS the modification operation of redaction is considered a public operation, i.e., involves no secret keys.

[728] The works cited are n° 24 [392], n° 43 [143], and n° 19 [133].

[729] Namely, the following joint works are cited n° 19 [133], n° 43 [143], n° 24 [392], and n° 32 [137].

[730] Shortcoming 5: Limited control of the structural integrity protection (see Sec. 12.5).

[731] The American English spelling from original work is retained for terms and quotes.

11.10. RSS: Algorithmic description

A redactable signature scheme consists in general of the following set of mathematically efficient[732] algorithms. Since the exact instantiation of these algorithms differs strongly between the various existing schemes, this section of the thesis limits the description to the input and output of each algorithm.

The set of algorithms present in this section has been formalised by *Brzuska et al.*, initially for tree-structured documents [66].

As before, in the following the entity of the signer always denotes the original signer of a given message m and the sanitizer denotes an entity that is authorized to sanitize at least one block of this m. Note, this thesis uses an indexed list like notation for RSSs without loss of generality, i.e., in general an RSS can be defined to work on any form of message. Regardless of its structure or even on fully or partially unstructured message it is assumed that the actual RSS defines how the message is decomposed into blocks.

Definition 169 : Redactable Signature Scheme

Any RSS consists of at least four efficient (PPT[733]) algorithms:

$$\mathsf{RSS} := (\mathsf{KGen}_{sig}, \mathsf{Sign}, \mathsf{Redact}, \mathsf{Verify}) .$$

Note, all algorithms may output $\perp$ in case of an error.

Key Generation: *There is a key generation algorithm for the Signer, which creates one pair of keys consisting of a private key and the corresponding public key:*

$$(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda) .$$

The security parameter is denoted as λ.

Signing: *The Sign algorithm takes as input the security parameter λ, a message $m = (m[1], \ldots, m[\ell])$, with each block $m[i] \in \{0,1\}^*$, the signer's secret key sk_{sig}, as well as a description ADM of the admissibly redactable blocks. ADM is a set containing just those blocks' indices which are modifiable, i.e., redactable. It outputs a signature σ and the message (or $\perp$ on error):*

$$(m, \sigma) \leftarrow \mathsf{Sign}(1^\lambda, m, \mathsf{sk}_{sig}, \mathsf{ADM}) .$$

This thesis assumes that ADM is always recoverable from any signature $\sigma \neq \perp$ by the sanitizer.

Redacting: *Algorithm Redact takes the security parameter λ, a message $m = (m[1], \ldots, m[\ell])$, with each block $m[i] \in \{0,1\}^*$, a modification instruction MOD, a signature σ, the signer's public key pk_{sig}. It modifies the message m according to the modification instruction MOD, which contains a set of block identifiers, i.e., $(j_0, \ldots, j_k)$, indicating those blocks that shall be redacted. The algorithm Redact generates a new signature σ' for the modified message $m' \leftarrow \mathsf{MOD}(m)$. Then Redact outputs m' and σ' (or $\perp$ in case of an error):*

$$(m', \sigma') \leftarrow \mathsf{Redact}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}) .$$

Note, the algorithm Redact requires no secret keys.

Verification: *The Verify algorithm outputs a decision as a bit $d \in \{\texttt{true}, \texttt{false}\}$ indicating the validity of the signature σ for the message m with respect to the Signer's public key pk_{sig}. `true` stands for a valid signature, while `false` indicates an invalid signature.*

$$d \leftarrow \mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}) .$$

[732] There are two efficiency notions in this work: First, the mathematic efficiency, and second, the "prototype" efficiency. The latter is interested in the impact in run-time that occurs and can be measured when comparing actual implementations.

[733] Probabilistic polynomial-time (PPT); see Definition 11 for definition.

11.11. RSS: Correctness properties

This thesis assumes that an RSS is correct. In a nutshell, the correctness properties for RSS require that every genuinely signed message must verify (signing correctness) and that every valid message that is genuinely redacted again must verify (redaction correctness). The formally defined signing correctness for SSS by *Brzuska et al.* in [64] was stated in Definition 145, but holds in the same manner for RSS. For readability of this section it is restated again.

The *sanitizing correctness* defined by *Brzuska et al.* [64] for SSS, stated in Definition 146, cannot be directly re-used, as RSS lacks the notion of the Sanitizer's key. For RSS *redaction correctness*, that was formalised for tree structured data as *cutting correctness* in [66], into this document's harmonised notation. Hence, this thesis defines *redaction correctness* for RSS as follows:

Definition 170 : Redaction correctness

> *All genuinely redacted messages (m, σ) are accepted as valid by* Verify*. For any security parameter λ, any key pair $(\mathsf{sk}_{sig}, \mathsf{pk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda)$, any message $m \in \{0,1\}^*$, any σ with $\mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}) = \mathtt{true}$, any* MOD *matching* ADM *from σ, and any pair $(m', \sigma') \leftarrow \mathsf{Redact}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig})$ results in:*
>
> $$\mathsf{Verify}(1^\lambda, m', \sigma', \mathsf{pk}_{sig}) = \mathtt{true}.$$

Analog to the work of *Gong et al.* for SSS, which requires that every SSS allows ADM to always be correctly recoverable from a valid message-signature pair (m, σ) [216, 427], this thesis also requires this for RSS whenever not all blocks are admissible. See the definition *ADM recovery correctness* given in Section 148 for details.

This thesis also requires algorithms to efficiently identify all the blocks $m_i \in m$ from a given message-signature pair (m, σ), *ease of decomposition* as defined in Definition 149.

11.12. RSS: Sequential executions of Redact

In general, the Redact algorithm can be executed again on already redacted data, thus this section discusses briefly the sequential execution of the Redact algorithm.

The sequential execution is already considered in the formalised definition given by *Brzuska et al.* [66] that the thesis bases upon. In [66] the Redact algorithm was called sCut and applying the leaf-cutting algorithm sCut subsequently allowed removing complete subtrees [66]. Taking the example of sCut also shows that subsequent application does not change each individual algorithm's execution behaviour: So if one execution of sCut does not allow redacting non-leaves or the re-locations of subtrees then also a sequential execution of sCut must not allow this.

Assuming a private RSS: Given the knowledge that m_i was redacted it is not possible to deduce from if this was done in the last iteration of the Redact algorithm or if there was a previous execution of Redact which affected m_i. This is the same as for SSS, see Sec. 11.5. Even if a non transparent RSS can leave an ■, the Verifier can just detect the existence of a redaction, i.e., by seeing an ■ symbol that this block was subject to a redaction. This may even be visible for every redacted block, if the redaction is not length-hiding. However, with a private RSS the Verifier is not able to conclude the order of applications of each sequentially applied Redact algorithm. Thus, the Verifier just sees the sum of all previous, including the latest, consecutive applications of Redact on the message. This includes even corner cases such as re-ordering the Redact executions, e.g., instructing a subtree removing Redact algorithm to remove children after a previous removal instruction had already removed the subtree starting at the children's parent would result in some errors but the final outcome of all sequential Redact executions would still be a tree without the subtree of the selected parent.

This is in line with the expected behaviour of subsequent overlapping authorized modifications to a document: "Every piece of data is as the last authorized modifier left it." was itself a quote that *Schneier* [433] used when presenting the notion of integrity [433, p. 122][734].

11.13. RSS: Existing security properties in harmonised notation

This section gives the state-of-the-art security properties for RSS in the notation used in this thesis. The cryptographic properties are as follows:

- Unforgeability (Sec. 11.13.1), which is closely related to
- Immutability (Sec. 11.13.2),
- Privacy (Sec. 11.13.3), and
- Transparency (Sec. 11.13.4).

Note, *accountability* has not been defined explicitly for RSS in the existing works, i.e.,it was not defined and discussed in *Brzuska et al.* [66], *Steinfeld et al.* [457], nor *Johnson et al.* [273]. However, accountability is relevant for the attribution of legal probative value, as the discussion in Sec. 7.4 and technical requirements like Requirement 4.1[735] shows. This thesis proposes accountability notions for RSS and also constructs a scheme that offers its public form: The need for accountability has been raised several times and finally introduced as Shortcoming 3[736]. The scheme denoted $pubacc\mathcal{RSS}$ (see Sections 14.13 and 14.14) fulfils the proposed framework for accountable RSS (ARSS) that is presented in Sec. 14.1.4.

11.13.1. Unforgeability for RSS

This property was formalised by *Brzuska et al.* in [66] for tree-structured documents.[737] The high-level definition of unforgeability can be captured for RSS as follows:

Definition 171 : Unforgeability for RSS (high-level)

Unforgeability *guarantees that a valid signature on a message can only be generated*

- *by the Signer with access to sk_{sig} corresponding to pk_{sig}, or*
- *by redacting a message-signature pair where the signature is valid under pk_{sig} and where all redactions are conforming to the Signer-endorsed modification policy (ADM).*

Note 1 All valid redactions are excluded from being forgeries.

Note 2 Unforgeability is as strong as possible; it shall explicitly prohibit unauthorized re-ordering of blocks of structured data, as well as mix-and-match attacks.

Note 3 Redaction is a public operation and requires no secrets.

Unforgeability assures that third parties cannot produce a signature for a "fresh" message without having access to the private signing keys. Fresh means the message has not been signed by the Signer and can not be produced from a message signed by the Signer through admissible redactions. This is explicitly explained in Note 2. This Note follows from the deficiencies found in Sec. 12.2 and aims to make RSS become as strong as possible RSS, by definition, offer public redaction. Hence, every party can assume the role of the Sanitizer. The unforgeability notion of RSS can be formulated to be similar to the unforgeability requirements of standard signature schemes [211] as already observed by [66]: The difference is that for RSS all valid redactions are excluded from the set of forgeries of that signed message.

[734] However, *Schneier* [433] did not provide the source of that quote.

[735] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1).

[736] Shortcoming 3: No, or no explicit public form of accountability (see Sec. 12.3).

[737] Note, the scheme by *Ahn et al.* published a year later achieves a less common notion of selective unforgeability [5].

11.13.1.1. Unforgeability for RSS in array notation

In the following this property will be presented in array notation, i.e., the i-th block of message m is denoted as $m[i] \in m$.

Definition 172 : Unforgeability for RSS

> *An RSS is **unforgeable**, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$ given in Fig. 57 returns* 1*, is negligible (as a function of λ).*

```
Experiment $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$
  $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk}, \cdot)}(1^\lambda, \mathsf{pk})$
    for $i = 1, \ldots, q$ let $m_i$ denote the adaptive queries to the oracle Sign
  return 1, if
    $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}) = \mathtt{true}$ and
    $m^* \notin \bigcup_{i=1}^{q} \mathsf{span}_{\vdash}(m_i)$
```

Figure 57. Unforgeability Experiment for RSS

In the unforgeability game given in Fig. 57 the adversary can use the Sign oracles to adaptively generate genuinely signed messages. The adversary has no access to the secret key generated as a first step. To model a breach of unforgeability, the adversary algorithm $\mathcal{A}$ must be able to compute a signature, which is valid under the public key corresponding to the secret key fixed at the beginning, on a message not previously queried from an oracle. The queries to the oracle are denoted by an index running from 1 to q. Hence, the adversary wins if $\mathcal{A}$ can output a *fresh* message and its corresponding modification policy with a signature valid under the fixed public keys of Signer and Sanitizer. Here, fresh means that the message is not equal to a message for which the adversary queried the oracle or that it cannot have been created by redaction from a message for which the adversary queried the oracle. This thesis assumes that the authorized modifications ADM can be recovered from a valid message-signature pair. The set of messages the adversary, and everyone else, can create by one or consecutive executions of the public algorithm Redact on a previously signed message m is denoted as $\mathsf{span}_{\vdash}(m)$. In other words, when the adversary is in possession of a validly signed m then $\mathsf{span}_{\vdash}(m)$ denotes "the set of signed messages in [the adversary's] possession" [105], as the adversary can create them by none, one or more redaction(s).

11.13.1.2. Unforgeability for RSS in set-like notation

Like all properties, the unforgeability can also be presented using a the set-like notation. As this property might be easier to understand in this notation, its experiment is next stated in set-like notation. Definition 172 would be unchanged, just referring to Fig. 58 instead of to Fig. 57:
An RSS is **unforgeable**, if for every efficient (PPT)[738] adversary $\mathcal{A}$ the probability that the game depicted in Fig. 58 returns 1, is negligible (as a function of λ).

```
Experiment $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$
  $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $(\mathcal{S}^*, \sigma_{\mathcal{S}}^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk}, \cdot)}(\mathsf{pk})$
    for $i = 1, 2, \ldots, q$
      let $\mathcal{S}^i$ index adaptive queries to the oracle Sign
      and let $(\mathcal{S}^i, \sigma_{\mathcal{S}}^i)$ denote the answers from the oracle
  return 1, if
    $\mathsf{Verify}(1^\lambda, \mathcal{S}^*, \sigma_{\mathcal{S}}^*, \mathsf{pk}) = \mathtt{true}$ and
    $\forall i, 1 \leq i \leq q : \mathcal{S}^* \nsubseteq \mathcal{S}^i$
```

Figure 58. Unforgeability Experiment for RSS using a set-like notation

[738] Efficient here means that the algorithm runs in probabilistic polynomial time (PPT) in the size of the security parameter λ.

No one should be able to produce a valid signature on a set $\mathcal{S}^*$ verifying under pk with elements outside the transitive closure of any set $\mathcal{S}$ received, without having access to the corresponding secret signing key sk. That is, even if an adversary can adaptively request signatures on different sets, it must remain unlikely to forge a signature for a new set not queried, i.e., $\mathcal{S}^* \not\subseteq \mathcal{S}^i$ for all $\mathcal{S}^i$ that were queried.

11.13.2. Immutability for RSS

This property exists for SSS and is described in Sec. 11.6.2 as follows:

Definition 153 : Immutability for SSS (high-level)

> ***Immutability*** *prevents a Sanitizer from deriving a verifying signature after modifying any block(s) not marked as admissible for this Sanitizer by the Signer.*

Note that the state-of-the-art schemes usually considered all blocks that were signed by an RSS as being also admissible to redaction. However, a redaction policy, i.e. ADM, can get described for RSS in an analog way as for SSS. Also in an RSS the Sanitizer must not be able to redact a given message in a malicious way. In particular, it must **only** be able to alter **admissible** blocks. Hence, also deleting non-admissible blocks or appending blocks must be prohibited. By use of notation[739], the experiment in Fig. 59 checks that a winning adversary's message is (1) validly signed and (2) could not be created by a valid redaction; the set of which are denoted as $\mathsf{span}_{\vdash}(m_i, \mathsf{ADM}_i)$.

Definition 173 : Immutability for RSS

> *An RSS is immutable, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment depicted in Fig. 59 returns* 1*, is negligible (as a function of λ).*

```
Experiment $\mathsf{Immutability}^{\mathsf{RSS}}_{\mathcal{A}}(\lambda)$
    $(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^\lambda)$
    $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk}_{\mathsf{sig}}, \cdot)}(\mathsf{pk}_{\mathsf{sig}})$
        for $i = 1, 2, \ldots, q$ let $(m_i, \mathsf{ADM}_i)$ index adaptive queries to the oracle Sign
    return 1, if
        $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathsf{sig}}) = \mathtt{true}$, and
        $\forall i \in \{1, 2, \ldots, q\} : m^* \notin \{\mathsf{span}_{\vdash}(m_i, \mathsf{ADM}_i)\}$
```

Figure 59. Immutability Experiment for RSS

11.13.3. Standard privacy for RSS based on *Brzuska et al.*

Informally, the property of privacy allows hiding the original un-sanitized contents once the block is sanitized.

EXAMPLE

Example: A third party cannot retrieve any additional information besides what is given in the message-signature pair from the redacted message and the signature alone [64]. For a tree-based structure, the adversary would be given a subtree with a signature σ and two possible source trees $T_{j,0}$ and $T_{j,1}$. If the scheme is standard private, then no one should be able to decide from which source tree the subtree stems from.

All privacy related properties try to minimise information leakage through information which can be deducted from the sanitized signature. Mechanisms for digital signature generation work regardless of the semantic contents of the document, hence they do not restrict the semantics of the message.[740] Hence, also information leakage by means of the modified message itself is out of scope as this cannot be prevented by a signature scheme.

EXAMPLE

For example, assume a set of strings is signed using an RSS for sets. Let the message be a set of strings: $\mathcal{S} = \{"org: A, B, C", "A", "B", "C"\}$. Then a redaction of the element containing $"A"$ seems not to remove the information about "A" semantically from the set.

[739] See Sec. 3.5 for more notations around ADM and MOD.

[740] See "Note that we do not consider semantics of the document." [349].

This definition is similar to the standard indistinguishability notation for encryption schemes [66] and it is the same as from SSS (see Definition 155); restated for readability:

Definition 155 : Standard Privacy

> *A* ***private*** *scheme prevents* Verifiers *from recovering any information (esp. the original value) about block(s) of m, which are no longer in the sanitized block(s) of m', through a valid signature σ' over m'.*
>
> *Note: Information leakage through the semantic content of the modified message m' itself, which is given to the adversary, is out of scope.*

Privacy for RSS has been formalised by *Miyazaki et al.* [347, 349] (see also Sec. 9.1.1) and also by *Brzuska, Busch, Dagdelen, Fischlin, Franz, Katzenbeisser, Manulis, Onete, Peter, Poettering, and Schröder* [66] (see also Sec. 9.1.5). Following the latter which was formalised for sets of blocks and all elements being admissible leads to the following:

Definition 174 : Standard Privacy for RSS

> *An* RSS *is* ***private****, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment* $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$ *given in Fig. 60 returns* 1 *is negligibly close to* $\frac{1}{2}$ *(as a function of λ).*

```
Experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$
  $(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathsf{sig}},\cdot),\mathsf{LoRRedact}(1^\lambda,\cdots,\mathsf{sk}_{\mathsf{sig}},b)}(1^\lambda, \mathsf{pk})$
    where oracle LoRRedact on input $m_0, \mathsf{MOD}_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}_0, \mathsf{ADM}_1$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}_0$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}_1$, return $\perp$
      if $\mathsf{ADM}_0 \not\subseteq m_0$ or $\mathsf{ADM}_1 \not\subseteq m_1$
        or $\mathsf{MOD}_0(\mathsf{ADM}_0) \not\subseteq \mathsf{MOD}_0(m_0)$ or $\mathsf{MOD}_1(\mathsf{ADM}_1) \not\subseteq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}_0) \neq \mathsf{MOD}_1(\mathsf{ADM}_1)$, return $\perp$
      let $(m_i, \sigma_i) \leftarrow \mathsf{Sign}(1^\lambda, m_b, \mathsf{sk}_{\mathsf{sig}}, \mathsf{ADM}_b)$
      return $(m', \sigma') \leftarrow \mathsf{Redact}(1^\lambda, m_i, \mathsf{MOD}_b, \sigma_i, \mathsf{pk}_{\mathsf{sig}})$.
  return 1, if $a = b$
```

Figure 60. Standard Privacy Experiment for RSS based on *Brzuska et al.* [66] including additions from *Krenn et al.* [294]

A third party, e.g., the Verifier, should not be able to gain any knowledge about redacted blocks without having access to them. The adversary can choose two tuples (m_0, MOD_0) and (m_1, MOD_1). A redaction instruction of MOD_0 applied to the message m_0 is required to result in the same output message as redacting m_1 according to MOD_1, i.e., $\mathsf{MOD}_0(m_0) = \mathsf{MOD}_1(m_1)$. The two tuples are input to a "Left-or-Right Redact" oracle which randomly chooses one of the two inputs, i.e., (m_b, MOD_b), and first signs and then redacts it. The internal results of the initial signing (m_i and σ_i) are kept secret.

Trivial ways to win the game are excluded. This is described in four abort conditions: First it is checked if both modifications of the supplied messages would result in the same redacted message ($\mathsf{MOD}_0(m_0) = \mathsf{MOD}_1(m_1)$). The following second and third checks remove the cases where the input would introduce errors to the algorithms due to an unauthorized request for modification, which could help to decide which message was chosen for input to the algorithms. The fourth is carried over from a recent discussion for SSS by *Krenn et al.* [294]; it ensures that the supplied policies ($\mathsf{ADM}_{\{0,1\}}$) before and after the modification match the respective message. The latter, i.e., $\mathsf{MOD}_0(\mathsf{ADM}) \neq \mathsf{MOD}_1(\mathsf{ADM})$, means that if the scheme has the potential to modify ADM that after the modification the ADM' of the resulting redacted message m' shall be the same regardless what input message was used for the sanitization. This was not found in the original game in [66]. It is added to cater for schemes that offer consecutive redaction control (see Sec. 14.5.1.1), which is then modelled to be an authorized modification of ADM.

The adversary wins, if it can decide (with non-negligible probability) which pair was used to as input to create the oracle's output. This is similar to the standard indistinguishability notion for encryption schemes.

11.13.4. Transparency for RSS

Definition 175 : Transparency for RSS

> *An RSS is **transparent**, if for any efficient adversary $\mathcal{A}$ the probability that the experiment given in Fig. 61 returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ).*

Transparency prohibits the adversary to decide whether a signature was generated using Redact or Sign. In the experiment the adversary chooses the message m to be signed and also the target outcome of the message's redaction by providing MOD. To prohibit trivial attacks the intended MOD needs to comply with the policy set by the also provided ADM. The Redact/Sign oracle then chooses randomly if it outputs the signature generated by signing the supplied message and then redacting it (case $b = 0$) or if it outputs a fresh signature using Sign on the modified message's contents. To win, the adversary has to identify with a non negligible probability if a signature over a supplied message was generated by Redact or Sign.

```
Experiment $\mathsf{Transparency}^{\mathsf{RSS}}_{\mathcal{A}}(\lambda)$
    $(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^{\lambda})$
    $b \leftarrow \{0,1\}$
    $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^{\lambda},\cdot,\mathsf{sk}_{\mathsf{sig}},\cdots),\mathsf{Redact/Sign}(1^{\lambda},\cdots,\mathsf{sk}_{\mathsf{sig}},b)}(1^{\lambda}, \mathsf{pk}_{\mathsf{sig}})$
        where oracle Redact/Sign on input of $m, \mathsf{MOD}, \mathsf{ADM}$:
            $\sigma \leftarrow \mathsf{Sign}(1^{\lambda}, m, \mathsf{sk}_{\mathsf{sig}}, \mathsf{ADM})$
            if $\mathsf{MOD} \not\subseteq \mathsf{ADM}$, return $\perp$
            $(m', \sigma') \leftarrow \mathsf{Redact}(1^{\lambda}, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{\mathsf{sig}})$
            if $b = 1$: $\sigma' \leftarrow \mathsf{Sign}(1^{\lambda}, m', \mathsf{sk}_{\mathsf{sig}})$
            return $(m', \sigma')$
    return 1, if $a = b$
```

Figure 61. Transparency Experiment for RSS

Transparency is not mandatory for RSS: Transparency is a stronger privacy property, i.e., Transparency ⇒ Privacy [66]. Transparency is important, if the existence of an occurred redaction must be hidden to increase the privacy. This is required, if any document sanitization whatsoever leads to disadvantages of any party involved. [64]. Still, *Brzuska et al.* show that the property of transparency is not mandatory to achieve a private RSS [66] (see also the overview of relations between properties in Sec. 11.14).

Transparency would hinder the detection of subsequent modifications by Verifiers on their own; this is a violation Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) as given in Req. 6. In line with the goal to increase the probative value this thesis will trade cryptographic transparency for the property of Non-interactive Public Accountability. Dropping transparency is more favourable from a legal perspective as the analysis of the impact on legal certainty highlights (see Subsections 6.4.2 and 6.4.5).

Recently, an even stronger transparency notion has been described by *Krenn, Samelin, and Sommer* [294]. However, this strengthening is not required as already the standard transparency negatively affects the legal value as it does not offer non-interactive public accountability (see Sec. 7.3.2 or discussions in Sec. 6.4.5).

11.14. RSS: Relations between RSS security properties

For some RSS properties relations have already been mentioned when the property was introduced. *Brzuska, Busch, Dagdelen, Fischlin, Franz, Katzenbeisser, Manulis, Onete, Peter, Poettering, and Schröder* [66] give the relations of the properties Transparency and Privacy [66]. The other property of RSS is unforgeability, which is independent. Note, the work on those properties by *Brzuska et al.* [66] was done for schemes working on tree structured data, but they can be simply carried over to other data structures.

The proofs of the relations for these state-of-the-art security properties are not restated in this section, but the interested reader is referred to the original source. Further to those relations, it is important for the probative value to achieve public forms of accountability while retaining confidentiality; this is stated as Theorem 3 for RSS.

Definition 176 : Transparency $\Rightarrow$ Privacy

> *"Any transparent structural signature scheme is also private." [66]*

Definition 177 : Privacy $\not\Rightarrow$ Transparency

> *"Suppose that there exists a private structural signature scheme. Then there exists a private scheme which is not transparent." [66]*

Definition 178 : Unforgeability is independent

> *"It is clear that unforgeability does not follow from privacy (and thus not from transparency). [...] Vice versa, it holds that unforgeability implies neither privacy, nor transparency [...]" [66]*

Following the focus of the thesis it is important to gain similarity to CDSS. For this sake, a mechanism should be able to offer privacy, but no transparency in order to achieve non-interactive public accountability. In particular its important for the further discussions in this thesis that $\neg$ *transparency* $\not\Rightarrow$ $\neg$ *privacy*.

Theorem 3 : $\neg$ Transparency $\not\Rightarrow$ $\neg$ Privacy

> *Assume that there exists a secure sanitizable signature scheme that is not transparent. Then there exists a sanitizable signature scheme which is immutable, unforgeable, sanitizer-accountable and signer-accountable, and private.*

Proof 3

> *This follows from the separation of transparency as stated and proven by Brzuska et al. [66]: Transparency is stronger and implies privacy (see Definition 176), but "[...] not all private [...] schemes are also transparent [...]" Brzuska et al. [66] (see Definition 177).* □

12 — Shortcomings of state-of-the-art SSS and RSS with respect to the requirements

Overview of Chapter 12

This thesis contributes by strengthening or increasing the level of detail of the existing security properties. With the enhancements the schemes achieve the goals which are desired in a setting that was distilled from the legal and economic requirements (see Chapter 7 and Chapter 10). This chapter lists major shortcomings with respect to the requirements. Thus with existing schemes it is impossible or problematic to achieve the desired goals. The existing security properties have already been presented in Chapter 11 — in a harmonised notation.

The existing SSS and RSS have been analysed for technical requirements necessary for a legally recognisable protection against recovery (cryptographic privacy) and for an increased probative value (integrity protection and accountability):

1. **Cryptographic privacy**: If a scheme or definition is not fulfilling the required level of privacy, then they are deemed to be unusable as they fail to fulfil Requirement 9[741];
2. **Integrity protection**: If a scheme or definition is not fulfilling the required level of integrity, especially $\geq$1CD (see Chapter 6), then they are deemed to be unusable as they fail to fulfil Requirement 3[742];
3. **Accountability**: If a scheme or definition is not fulfilling the required level of accountability, especially not fulfilling non-interactive public accountability (see Requirement 4) like cryptographically transparent schemes, then they are deemed at least problematic as they fail to fulfil recommended parts of Requirement 4[743].

Additionally, the existing SSS and RSS have been analysed regarding their:

4. **Flexibility of sanitization and redaction**: If a scheme is not allowing fine-grained control of the scope of protection and is too limited in the subsequent modifications that the Signer can authorize, then they are deemed to be too specific to attract attention for a widespread usage. Moreover, the runtime overhead should not make the scheme become impractical very fast, because otherwise the scheme will not be able to be used for larger messages or data sets and thus might not be attractive for a widespread use.

For all these shortcomings this thesis proposes new security properties to counter them and a suitable scheme that fulfils the new properties: Three new SSS are proposed in Chapter 13 and five new RSS are proposed in Chapter 14. Formal security proofs are given for them.

[741] Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength) (see Sec. 10.1).

[742] Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification) (see Sec. 7.3).

[743] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

12.1. Shortcoming 1: Not all schemes achieve sophisticated levels of privacy

Privacy, as captured in Requirement 9, is one strong reason why the application of malleable signatures is viable. Several publications emerged after the initial papers introducing RSS and SSS, of which some introduced their own definitions of cryptographic privacy properties, including transparency and unlinkability properties. The formalisation of the privacy properties from *Brzuska* et al. [64–66] also showed that proposed existing schemes were not fully privacy preserving according to their model.

EXAMPLE

As an example of a scheme that is not private under this definition take an early scheme proposed by *Pöhls* [375] (see Appendix A publication n^{o} 4). It is constructed from a Merkle hash tree (MHT) where the blocks of the message are distributed as the content of the leaf-nodes. The node's content is not padded with a random number before calculating the leaf's content, i.e., the construction just hashes the block's content to become the MHT's leaf-node. A positive verification will require either the original content of the node or the node's intermediary hash-values from the MHT, i.e., redacting a leaf-node's content exchanges the content with the content's hash-value. This is not enough, and the scheme fails the privacy notion of *Brzuska et al.* [66]: Due to a predictable hash-function, the adversary in the privacy game can distinguish different content in one block, e.g., $m^* = "A||Z"$ and $m^{**} = "B||Z"$ shall be redacted into $m' = "Z"$. The adversary can calculate the hash-value for the position of the $"A"$ or the $"B"$ respectively. It will be able to identify which input was hashed to produce the oracle's output by observing the hash-value, i.e. $\mathcal{H}("B")$ or $\mathcal{H}("A")$, that is placed in the signature to substitute the redacted content. Thus, the standard notion of privacy based on *Brzuska et al.* [66] can not be met by schemes where the output of a non-randomised one-way-function over redacted content remains in the redacted signature.

But even using random numbers is not always enough. Using a probabilistic attack, *Brzuska et al.* broke schemes that claimed to even[744] offer transparency, but did fail to do so and then also failed to provide privacy [66]. Especially, schemes which are not transparent were subject to such a 'staged' attack as the adversary's ability to see that an element has been redacted — which breaks transparency — can be used to attack privacy. That schemes which do not achieve transparency have shown to often offer no strong privacy guarantees either was already noted in [348] and [296]. If privacy is not upheld, a Verifier can make statements about the original message m, which contradicts the intention of an RSS [66]. That this property must be upheld and must withstand these kind of attacks got captured as Requirement 9[745]. Therefore the formalisation of the privacy-related properties from *Brzuska et al.* got selected as the basis for this thesis's property of standard privacy as defined in Sec. 11.6.3 for SSS and in Sec. 11.13.3 for RSS. Standard privacy as defined in *Brzuska et al.* [64] only requires that the adversary supplied inputs "are mapped to the same modified message" [64] by the oracle. In the original game of privacy for SSS by *Brzuska et al.* [64], and also in later works [67], the adversary is allowed to input of the pairs $(m_0, \mathsf{MOD}_0, \mathsf{ADM})$ and $(m_1, \mathsf{MOD}_1, \mathsf{ADM})$. Especially, in the original definition the adversary's ADM is the same for both messages [64, 67]. Note, *Brzuska et al.* call a scheme secure if it is unforgeable, immutable, private, Signer-/Sanitizer-accountable and private[746].

Some schemes that have been shown to have security weaknesses in *Brzuska et al.*'s model are summarised in Tab. 9. Note that Tab. 9 is by no means exhaustive.

Due to the attacks possible, schemes that have not been proven to fulfil at most standard privacy (see Definition 156 and Definition 174) are deemed to not fulfil Requirement 9. Hence, those schemes are not considered suitable for any application that seeks to protect the confidentiality of trade secrets or the privacy of personal data by sanitization or redaction.

[744] 'Even' because transparency implies standard privacy.

[745] Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength) (see Sec. 10.1).

[746] They state for RSS that "[...] for various application examples privacy is in fact sufficient, namely in all cases, where the receiver already expects partly sanitized documents." [66]. For SSS the authors state a "[...] scheme is [...] unforgeable and private and thus a secure sanitizable scheme" [66].

Scheme	Attack given in	Note
Steinfeld et al. [455]	*Brzuska et al.* [66]	*Steinfeld et al.* [455] "[...] do not require hiding the amount of data removed from the document, i.e., one is able to derive the lengths of the removed strings, or where these were removed from." [66].
Ateniese et al. [12]	*Brzuska et al.* [64]	needs tags and chameleon hash-function must be coll.-resistant under random-tagging attacks
Yuen et al. [508]	*Agrawal et al.* [4]	"The scheme in [508] does not provide transparency." *Agrawal*
Idalino et al. [238]	Sec. 12.1.1	not private; leaks hash-value of original m
Izu et al. [264]	Sec. 12.1.2	leaks number of redactions as redacted signature still contains each redacted content's hash-value
Wu et al. [507]	Sec. 12.1.3, [295, 296]	no privacy
Kundu and Bertino [296]	*Brzuska et al.* [66], Sec. 12.1.4.1	not transparent due to ordered random numbers [66]; argued that is private because its transparent
Kundu and Bertino [295]	*Brzuska et al.* [66], Sec. 12.1.4.1, Sec. 12.1.4.2	"does not achieve transparency [...]" [66]; no privacy

Table 9. Selected existing schemes which have shortcomings by not achieving standard privacy (as defined in Definition 156 and in Definition 174) or have shortcomings with privacy related properties; for an explanation of the attacks please see the respective stated papers; for new attacks or more explanation see indicated subsections.

12.1.1. Problems of scheme by *Idalino et al.* towards privacy

The scheme proposed by *Idalino et al.* does not explicitly claim to protect the privacy. However, as the authors claim that partial data integrity mechanisms, like theirs, can be "solution for guaranteeing privacy protection" [238] the mechanism is in the following analysed for standard privacy. On signature generation, all the n message blocks, denoted B_j, are transformed directly using a standard collision-resistant hash-function[747] and then used in the matrix M. Assume one can detect with the demanded precision which blocks are not valid under the signature using the matrix, i.e., one learns which blocks have been subsequently modified; then the following steps can be taken:

> "Let $h_j \leftarrow h(B_j)$, $1 \leq j \leq n$. Use $(h_1, h_2, ..., h_n)$ and M to compute $T_1, ..., T_t$, as follows: for each row $1 \leq i \leq t$ compute c_i, which is the concatenation of the hashes h_j for j such that $M_{i,j} = 1$, and let $T_i = h(c_i)$. Calculate $h^B = h(B)$ and set $T = (T_1, T_2, ..., T_t, h^B)$. Compute $\sigma^T = \mathsf{S.Sign}(\mathsf{sk}, T)$ and output $\sigma = (T, \sigma^T)$."[748] [238]

The problem is that hashing also the blocks that have been removed leaves their footprint, i.e., the hash-value of the content, in the resulting σ. Namely, the complete message's hash-value h^B is supplied to the verifier.

This allows the privacy adversary to draw a relation between original input and redacted output: The adversary supplies two messages m_0 and m_1 to the $\mathsf{LoRSign}$ oracle of *Brzuska et al.*'s privacy game and then checks if the result would carry $h(m_1)$ or (m_0) as h^B in the signed T. This breaks privacy.

Idalino et al. make no special comment if and what needs to be additionally achieved by the underlying signature scheme, i.e., $\mathsf{S.Sign}$, to meet the standard privacy nor in which security model. It is out of scope of this thesis to discuss if the above could be fixed, e.g., in the random oracle model by adding random values as padding to each block.

In the supplied version the leak of the original message's hash-value, which is not randomised, in the signature is making them obviously miss standard privacy [66].

[747] They require that "[...] the hash-function used has the desired properties (no collisions) [...]" [238].

[748] Some notation adjusted from the original to avoid using * and $'$ which in this thesis are used for adversary supplied or subsequently modified values.

12.1.2. Problems of *Izu et al.* towards standard privacy

Note, following from the discussion on attacks on transparency in *Brzuska et al.* [66] it is assumed that hiding the number of redacted blocks might be considered a goal of privacy. The scheme by *Izu, Kunihiro, Ohta, Sano, and Takenaka* [265] has the following construction: The signature created by the Signer consists of a list of hash-values generated from each original block's content. When redactions occur, the content is overwritten or left-out and a new hash-value is put into the list of hash-values (denoted as H in [264]) replacing the original one. This new list of hash-values H' is then signed by the Sanitizer. However, the original signature of the Signer including the original list of hash-values H still needs to be provided to the Verifier[749]. This leaks the original content's hash-value even if it was redacted. This might be enough to achieve the authors' property of "[...] secrecey [sic] (no leakage of sanitized information) [...]" [264] which the authors found "[...] assured by the preimage resistance of the hash function." [264]. However, as the hash-value depends on the input and both the input as well as the list of hash-values is known to the adversary, the adversary can break the privacy game of *Brzuska et al.* [64][750] by providing two messages with different content in one block, e.g., $m^* = ''A||C''$ and $m^{**} = ''B||C''$ which shall be redacted into $m' = ''C''$. While the adversary can not obtain the internal signature generated inside the LoRSign oracle of *Brzuska et al.*'s privacy game it can apply the hash-function itself. As the hash-value is not randomised the adversary obtains hash-values belonging to $''A''$ or to $''B''$. With a high probability the hash-values are different. The adversary then compares them with the entries for H^* and H^{**} and thus identifies the input used and wins the privacy game with a probability that is non-negligible better than $\frac{1}{2}$.

12.1.3. Problem of *Wu et al.* towards standard privacy

The RSS scheme for trees by *Wu et al.* [507] relies on the *Merkle*-Hash-Tree-technique with standard cryptographic hash-functions like SHA-512. Hence, their scheme does neither hide redactions (transparency) nor preserve privacy as shown by [295, 296]: The adversary is able to break privacy by supplying the LoRSign oracle of *Brzuska et al.*'s privacy game with two different strings and then observing the remaining hash-value for that block within the Merkle hash-values and wins the privacy game with a probability that is non-negligible better than $\frac{1}{2}$.

12.1.4. Problems of *Kundu et al.* towards privacy

The schemes by *Kundu et al.* [295, 296] lacks provable transparency since they use ordered random numbers [66]: A probabilistic attack can be mounted to break their claimed transparency as shown by *Brzuska et al.* [66]. One may argue that *Kundu and Bertino* do not see their scheme in the context of RSS. However, they themselves state that they want to prohibit "leakage" due to structural information [296].

Transparency and privacy can actually be attacked in many ways. In the following, some of the known attacks on privacy and transparency on the schemes by *Kundu et al.* [295, 296] are stated[751]. Sec. 12.1.4.1 describes the probabilistic attack against transparency as described by *Brzuska et al.* [66]. Sec. 12.1.4.2 describes further attacks not published before. Those attacks got published as part of joint work with *K. Samelin*, *A. Bilzhause*, *J. Posegga*, and *H. de Meer Samelin et al.* [425] at the ACNS conference in 2012 (see Appendix A publication n° 10).

12.1.4.1. Randomised traversal numbers

The following attack has been discovered by *Brzuska et al.* in [66]: *Kundu and Bertino* propose three ways to randomise the traversal numbers in their work [296]:

Sorted Random Numbers: Generate $|V|$ random numbers and sort them.

Order-Preserving Encryption: Apply an order-preserving encryption scheme, e.g. [3], to the traversal numbers.

[749] "Then the sanitizer publishes the sanitized document and the sanitizer's signature s' in addition to the original signature s." [264].

[750] See Sec. 9.1.5.

[751] Some attacks can just be mounted on special revisions of the schemes by *Kundu and Bertino*. This is stated for each attack.

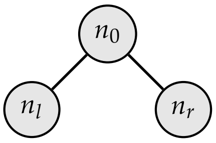


Figure 62. Sample tree with three nodes

Addition of Random Numbers: Assign the numbers to the nodes by taking the previous traversal number and adding a random offset.

Consider a tree with three nodes, i.e., the tree as depicted in Fig. 62. Assume the algorithm Θ outputs the traversal numbers assigned to the tree's nodes on input of the depicted tree T in Fig. 62 and a distribution Δ:

$$(r_l, r_r, r_0) \leftarrow \Theta(T, \Delta)\,.$$

Assume that Δ is the uniform distribution and let $\mu = \mathrm{E}(r_l)$ be the expected random number associated to n_r. Furthermore, assume that $\Pr[r_l = \mu] = 0$ for a sufficiently large space. Hence, the probabilities are $\Pr[r_l < \mu] = \Pr[r_l > \mu] = 0.5$. Therefore, one obtains $\Pr[r_r > \mu] \geq 0.75$, since r_r is the largest random number and for $r_r < \mu$, both r_l and r_0 must be smaller than μ. The statistical attack on transparency for the sample tree depicted in Fig. 62 works as follows: Transparency has been defined as a game where the adversary must guess a bit b. The game, on input of a the tree with three nodes and the instruction to remove n_l either returns a signature over a tree just containing n_0 and n_r or a signature which was generated by first signing the whole three nodes and then cutting off n_l [66]. To win the game, the adversary simply outputs 1, if $r_r \leq \mu$ and 0 otherwise. Note: $\Pr[r_r > \mu] \geq 0.75$ for a sanitized tree, while for a freshly signed tree the probability changes: $\Pr[r_r > \mu] = 0.5$. Hence, b can be guessed with a non-negligible probability. Therefore, the scheme by *Kundu and Bertino* is neither transparent, as already shown by [66], nor private, since the leaked information allows to make statements about the original message. This attack works for other distributions as well, while it is required that the adversary knows the distribution Δ it can be derived from the signing algorithm which might be known to the adversary[752].

12.1.4.2. Leakage through the structural signature G_T

The following attack description was published as part of the joint work with *K. Samelin*, *A. Bilzhause*, *J. Posegga*, and *H. de Meer* [425] (see Appendix A publication n° 10).

In the original paper [295], *Kundu and Bertino* introduce an additional hash-value which they name "structural signature", denoted as G_T. This "signature" is calculated as $G_T \leftarrow \mathcal{H}_s(\rho_1||c_1||\ldots||\rho_n||c_n)$. It is part of the calculation of each σ_i, i.e., $\sigma_i \leftarrow \mathcal{AGG}.\mathsf{ASign}(\mathsf{sk}, G_T||\rho_i||c_i)$. Obviously, to check the signature σ_T, G_T needs to be available to the $\mathsf{Verifier}$. The $\mathsf{Verifier}$ can calculate a hash-value G'_T on input of the tree T'. In particular, $G'_T \leftarrow \mathcal{H}_s(\rho_1||c_1||\ldots||\rho_{n'}||c_{n'})$, where $n' = |V'|$. Afterwards, the following equation is checked: $G'_T \overset{?}{=} G_T$. Following the definition of standard privacy (see Sec. 11.13.3), this allows the adversary to identify from a redacted tree between two un-redacted sources that were input to the $\mathsf{LoRSanit}$ oracle with high probability; this destroys the privacy protection.

This problem has partially been circumvented in [296] by salting G_T; in particular the Signer calculates: $G_T \leftarrow \mathcal{H}_s(\omega||\rho_1||c_1||\ldots||\rho_n||c_n)$, where ω is a nonce. As the adversary will not see the original signature[753], under the assumption that the signature is truly random and the hash-function is oneway, the adversary will not be able to attack this. However, this choice and the assumptions requires that $\mathcal{H}_s$ is modelled as a random oracle to ensure privacy [296]; this means the scheme is no longer secure in the standard model. In the revision introduced in [298], no structural signature is present, thus it does not inherit this problem. However, when omitting it completely there is no more document tag, which renders that revision of the scheme forgeable through a mix-and-match attack, this destroys its integrity protection capabilities as is shown in Sec. 12.2.1.2.

[752] Only the secret keys are assumed to be secret, not the algorithms or their parameters.

[753] the inner workings, in particular the signatures σ_i, are not given to the adversary [64].

12.1.5. Stronger notions of privacy

Apart from the notion introduced as standard privacy, there are stronger privacy properties like transparency (see Sec. 11.6.4 on page 265 for SSS and Sec. 11.13.4 on page 290 for RSS) and unlinkability (see Sec. 11.6.5 on page 269 for SSS and Sec. 14.3 on page 376 for RSS). This thesis itself presents some strengthening to the property of privacy in Sec. 15.1.3 and also strengthens the notion of unlinkability in Sec. 13.3. Additionally, advances have been published in joint publications which resulted from the work or are related to this thesis (see Appendix A). Independently, the notion of privacy has recently — in 2015 — been further strengthened by *Krenn et al.* [294] and by *Derler and Slamanig* [139]. *Ahn et al.* introduced a privacy-related notion termed context-hiding [5]. Even stronger privacy notions have then been introduced by *Attrapadung et al.* [15] and *Attrapadung et al.* [16]. However, the scheme by *Ahn et al.* only achieves the less common notion of *selective* unforgeability [5]. Moreover, [5, 15, 16] are limited to quoting, i.e., redactions are only possible at the beginning or at the end of a list. The afore defined standard privacy definition based on the work of *Brzuska et al.* [64] only considers outsiders as adversarial. However, one could require that even insiders, i.e., sanitizers, are not able to win the game. This strengthened definition is presented in Sec. 15.1.3 on page 433.

12.2. Shortcoming 2: No full protection of all aspects of integrity

This thesis differentiates between content, structural and referential integrity (see Sec. 7.1). An integrity mechanism shall clearly state the scope and strength of the integrity protection. If the protection is less, then the Signer must be clearly give his consent, hence the mechanisms strength needs to be known to the Signer. Tab. 10 presents some examples for schemes that have been shown to have security weaknesses in their offered integrity protection model.

Scheme	Attack given in	Note
Miyazaki et al. [348]	Sec. 12.2.2	for sets, but missing uniqueness check allows for duplication attacks
Kundu and Bertino [295, 296, 298]	Sec. 12.2.1	only transitive closure protection allows level-promotion attacks; mix-and-match attacks even break unforgeability

Table 10. Selected existing schemes where shortcomings in integrity protection or related properties are presented

12.2.1. Problems of *Kundu and Bertino* towards unforgeability and structural integrity

The first attack[754] presented in Sec. 12.2.1.1 shows that with the allowed flexibility of intermediate, i.e., non-leaf, node redactions the schemes by *Kundu and Bertino* [295, 296, 298] do not protect the structural integrity with regards to the nodes level (or depth) and father-child relation of nodes.

The second attack[754] presented in Sec. 12.2.1.2 shows that the scheme from [298] is susceptible to so-called mix-and-match attacks allowing the adversary to craft new validly signed tree-structured documents from several different signed documents; this violates the unforgeability property, as the new documents are not redactions but mixtures. With already two attacks on the integrity, this thesis concluded that all revisions, i.e. [295, 296, 298], are not secure enough to fulfil the integrity requirement.

12.2.1.1. Level promotion means reduced structural integrity protection on tree-structured data

The attack shows how to alter the semantic meaning of tree T by removing nodes which are not leaves of the tree. *Kundu and Bertino* [295, 296, 298] name non-leaf nodes or "intermediate nodes". This thesis adapts their nomenclature.

[754] The two attacks were timely disseminated and published as joint work with *K. Samelin, A. Bilzhause, J. Posegga*, and *H. de Meer* [425] at ACNS (see Appendix A publication nº 10) and jointly with the same co-authors in [426] at ISPEC (see Appendix A publication nº 13) in the year 2012.

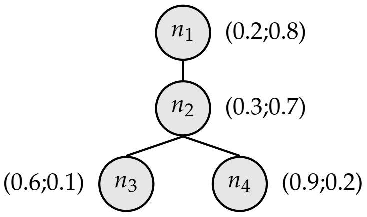


Figure 63. Original tree T with traversal numbers

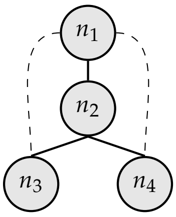


Figure 64. Transitive closure of T from Fig. 63, i.e. $\text{span}_{\vdash}(T)$

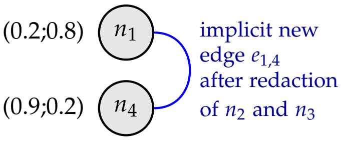

(0.2;0.8) n_1
(0.9;0.2) n_4
implicit new edge $e_{1,4}$ after redaction of n_2 and n_3

Figure 65. Redacting T from Fig. 63 into T'

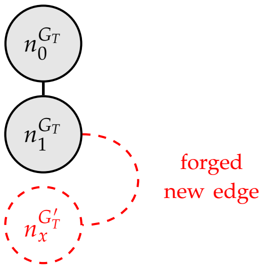


Figure 66. Mix and match attack

The scheme got introduced in [295] and revised in [296]. It builds upon the idea that a third party having the pre- and post-order traversal numbers of all nodes contained in a tree T is always able to correctly reconstruct T (see Fig. 63). Hence, signing each node $n_i \in T$ along with both numbers and the content is enough to protect T. To make the scheme transparent, these traversal numbers are randomised in an order-preserving manner, which does not have an impact on the reconstruction algorithm, which just checks for greater than relations [295, 296]. Thus, verification is straightforward by verification of the signatures on each given node — with one additional step: The Verifier has to check if all nodes are in the correct order using the traversal numbers. The authors argue that signing each node $n_i \in T$ along with both traversal numbers is enough to protect the tree's integrity. This, however, is not correct, as it leads to the problem that a Verifier is not able to determine whether a given edge existed in the original tree T, just if it *could* have existed in the tree's transitive closure (see Fig. 64). Removing intermediate nodes allows introducing new implicit edges in certain scenarios.

EXAMPLE

In the following example this attack is explained in detail: First the algorithm needs to compute the traversal numbers for the original tree in Fig. 63: The pre-order traversal of T will output $(1, 2, 3, 4)$, while the post-order traversal will output $(4, 3, 2, 1)$. Hence, the node n_1 has a structural position of $\rho_1 = (0.2; 0.8)$. For n_2, n_3 and n_4, this is done accordingly. The randomisation step may transform them into $(0.2, 0.3, 0.6, 0.9)$ and $(0.8, 0.7, 0.2, 0.1)$ respectively.

Assume a redaction of the intermediate node n_2 and the leaf-node n_3 takes place. This is depicted in Fig. 65. Note that this added a new edge $e_{1,4}$, which has not explicitly been present in the original tree T. However, for the redacted tree in Fig. 65, the traversal-numbers are still in the correct order. This implies that the tree $T^{\mathcal{A}} = (\{n_1, n_4\}, \{e_{1,4}\})$ is valid in terms of the signature. Hence, the signature verifies.

The resulting behaviour is problematic unless the Signer is consenting to the structural integrity protection of *Kundu et al.*'s scheme, knowing well that it is being limited to a protection of the transitive closure of the tree T, e.g. as depicted in Fig. 64. Nevertheless, content integrity is provided for each node.

***Kundu et al.*'s scheme introduced in [295] and revised in [296] offers only limited structural integrity (i.e. only ancestor-of).** Note, the scheme by *Kundu and Bertino* was the only existing RSS for trees at that time that explicitly allowed to redact non-leaf nodes. It was introduced in [295] and revised in [296]. The schemes described neither prohibit nor exclude the redaction of intermediate nodes. This thesis fully supports the authors claim that this is a useful property [296]. However, by its use of traversal numbers the construction given by *Kundu et al.* signs the transitive closure of the tree T, e.g., Fig. 64. This means that such a scheme only protects the ancestor relationship of nodes in T (e.g. Fig. 65 is a validly signed redacted tree). In turn, the structural integrity protection (see

Definition 115 given for Requirement 1.2 in Sec. 7.1.2) is limited and the Signer implicitly signs all edges of the transitive closure of T. Hence, signing each node $n_i \in T$ along with both numbers and the content is enough to protect the full structure, e.g. level, of all nodes in T. This leads to the problem that a Verifier is not able to determine whether a given edge existed in the original tree T, just if it *could* have existed in the transitive closure. As a result, signing a tree with this construction authorizes a third party to add edges to the tree T' that were not present in the original T. This is a limit to the structural integrity protection. The protection offered is weak and allows certain semantic attacks: Especially it allows moving existing nodes upwards within the levels of the tree. This attack was thus named "level-promotion" in the joint work [426] (see Appendix A publication n$^{\text{o}}$ 13).

Strictly speaking, this destroys full structural integrity of a tree, and therefore has a negative impact on unforgeability and integrity protection. Note, this attack is possible in all versions of the schemes by *Kundu and Bertino* [295, 296, 298]. One might argue, that it depends on the definition of the integrity protections scope and strength, if this behaviour can be considered as an attack. However, such powerful possibilities must be under the sole control of the Signer to avoid unwanted side-effects (see Requirement 2.1[755]) and it must be made explicit for the Signer to give his consent (see Requirement 2.2[756]). This limitation was, however, not explicitly stated in their original work. Additionally, the attack is applicable if one is able to redact parent nodes to allow distributing subtrees of a given tree. In the example, this would be the tree $T' = (\{n_1, n_2\}, \{e_{1,2}\})$. Obviously, this also leads to the same problem. One may argue that intermediate node redaction lacks application scenarios. This is not true.

EXAMPLE

Consider the following example: A tree's structure is implicitly describing the hierarchy within a company, e.g. employees are grouped in departments below an employee in the role of a department head. Allowing to remove intermediate nodes now permits to solely remove the node of the department head not needing to remove the whole subtree containing nodes representing the employee. This allows to redact until only a list of employees remains. Without intermediate node redaction, generating such a list by redaction is not possible.

Hence, there exist use cases, where being allowed to remove solely leaf-nodes, or iteratively whole subtrees, is not sufficient.

12.2.1.2. Mix-and-match attacks break unforgeability

The mix-and-match attack is an attack where the adversary creates a forgery by building a message from blocks which have all appeared in genuinely signed messages, but have never appeared together in a signed document — following the description from [455][757].

In the older revisions of the schemes by *Kundu and Bertino*, i.e., as proposed in [295, 296], each node $n_x^{G_T}$ is bound to the tree T^{G_T} due to the structural signature G_T, which is part of each signature. Hence, a node $n_x^{G_T}$ cannot be merged into another tree $T^{G'_T}$, which nodes $n^{G'_T}$ have been signed with a different tag: G'_T instead of G_T. However, in the newest revision [298], this tag is not present anymore. It is just aggregated onto the signature, but does not bind nodes to the tag, since it is not part of the signature generation of each node. In particular, $\sigma_i \leftarrow \mathcal{AGG}.\mathsf{ASign}(\mathsf{sk}, G_T||\rho_i||c_i)$, where $G_T := \emptyset$. Hence, an adversary $\mathcal{A}$ can merge nodes of different trees into a new tree $T^{\mathcal{A}}$, which contains nodes originally signed for $T^{G'_T}$ and T^{G_T}.

EXAMPLE

The following example highlights the attack: Fig. 66 on page 299 depicts the resulting forged tree $T^{\mathcal{A}}$ with three nodes. It can be forged from two validly signed trees $T^{G'_T}$ and T^{G_T}. W.l.o.g. the order preserving randomisation is omitted for clarity in the following description. Let $\rho_0 = (1, 3)$, $\rho_1 = (2, 2)$ be the traversal numbers of the two nodes of tree T^{G_T}; and let $\rho_x = (3, 1)$ be the traversal number of one node of $T^{G'_T}$.

[755] Requirement 2.1 (Mechanism MUST enable the Signer to control scope and strength of the integrity and authenticity protection).

[756] Requirement 2.2 (Mechanism MUST verifiably document Signer's consent for chosen integrity and authenticity protection).

[757] The attack's name was changed, but the description was only adapted following the original one given by *Steinfeld et al.* [455]: "[...] 'mixed subdocument attacks' (where the forged document contains submessages which have all appeared in signed documents but have never appeared together in a signed document)." [455].

The adversary $\mathcal{A}$ has access to each of the individual signatures $\sigma_{n_0}, \sigma_{n_1}, \sigma_{n_x}$ and the "blinding signatures" σ_{ω_T} and $\sigma_{\omega'_T}$.[758] Hence, the adversary can construct a signature $\sigma_{\mathcal{A}}$ on the tree $T^{\mathcal{A}}$ by calculating $\sigma_{\mathcal{A}} \leftarrow \mathcal{AGG}.\mathsf{AAgg}(\mathsf{pk}, \{\sigma_{n_0}, \sigma_{n_1}, \sigma_{n_x}, \sigma_{\omega_T}\})$ resp. $\sigma'_{\mathcal{A}} \leftarrow \mathcal{AGG}.\mathsf{AAgg}(\mathsf{pk}, \{\sigma_{n_0}, \sigma_{n_1}, \sigma_{n_x}, \sigma_{\omega'_T}\})$, where $T^{\mathcal{A}}$ consists of all three nodes mentioned. Running $\mathsf{TVerify}(T^{\mathcal{A}}, pk, \sigma_{\mathcal{A}})$ outputs the bit `true`. As a result, the attacker has successfully forged a signature for $T^{\mathcal{A}}$.

Such a mix-and-match attack can be applied as soon as two trees are signed with the same private key. Hence, the newest revision of the scheme by *Kundu and Bertino* [298] is forgeable. Note, the older revisions do not have this problem, as every node $n_x^{G_T}$ is bound to the tree T^{G_T} due to G_T, which is unique for each signed tree. However, the way the tag G_T was designed makes the scheme fall victim to an attack on privacy (see Sec. 12.1.4.2).

12.2.2. Problems of *Miyazaki et al.* towards structural integrity

The next attack[759] is against the expected structural integrity of the scheme by *Miyazaki, Hanaoka, and Imai* [348]:

The scheme is proposed to protect the integrity of data structured in sets. While this means that the ordering is not part of the structural integrity protection, the uniqueness of each set element could be assumed to be protected. However, *Miyazaki et al.* [348] forget to check during verification for the absence of multiple equal elements in the set. They assume that each element, resp. subdocument must be unique. However, within their verification algorithm this is not checked, hence their scheme is forgeable, i.e., by copying existing elements. As such, the Signer would actually authorize the creation of validly signed multi-sets. In favour assume the scheme from [348] would have been planned to protect the integrity of multi-sets. Still their scheme is not protecting the structure of multi-sets as an adversary would still be able to duplicate elements or remove existing duplicates. The scheme does not protect the number of occurrences of each set element at all. This is seen as a problem towards the protection of structural integrity for structured data in the structure of a set. Of course this is technically easy to fix by adding an additional check for uniqueness, but the question remains how to advertise the structural integrity provided by the original scheme from *Miyazaki et al.* [348].

Without a fix those problems are a hinderance, especially for the probative value. For the legal attribution of a high probative value it is of paramount importantance that the protective capabilities of the integrity mechanism are clearly communicated to the Signer in order to allow to argue that the Signer was aware of the authorization for subsequent edits it is giving. This is captured in this thesis at least in Requirement 1.2[760] and in Requirement 2.2[761].

12.3. Shortcoming 3: No, or no explicit public form of accountability

The definition of transparency (see Sec. 11.6.4 and Sec. 11.13.4) prohibits any public form of accountability. Accountability is seen as required for a secure MSS it can, however, be provided independent of transparency [64]. As discussed previously[762], public accountability here means that it is required that third parties can immediately decide whether a message has been issued by the Signer or the Sanitizer without further interactions or inputs. Transparent schemes require the Signer's secret key to run the additional algorithm Proof in order to resolve the question who is accountable. Some of the schemes proposed in the existing literature are not transparent, i.e., a Verifier can identify that a third party redacted something [64, 222, 273, 347, 455, 507]. However, as discussed in detail as Theorem 5 (see

[758] An adversary can always build an inverse of all signatures received.

[759] The following attack has been published in joint work with *K. Samelin, A. Bilzhause, J. Posegga*, and *H. de Meer* [426] at ISPEC 2012 (see Appendix A publication n° 13).

[760] Requirement 1.2 (Mechanism MUST protect structural integrity as requested by the Signer) (see Sec. 7.1.2).

[761] Requirement 2.2 (Mechanism MUST verifiably document Signer's consent for chosen integrity and authenticity protection) (see Sec. 7.2.2).

[762] See for example Sec. 6.4.4 or Sec. 6.4.5.

Sec. 13.1.3) not achieving transparency is not enough to achieve non-interactive public accountability. Notably, the construction given by *Brzuska et al.* [64] is not just not transparent but achieves the public form of accountability that this thesis proposes[763]. The property of public accountability was however not formalised or discussed further in [64].

Note, the property of transparency was initially not well defined in the work of *Ateniese et al.* [12]. A discussion on the details of the problems of the transparency definition is given in Sec. 11.6.4. This finding and its rectification was also published in the joint work with *Samelin and Posegga* [391] (see Appendix A publication n$^{\text{o}}$ 6). This thesis harmonised and refined the transparency definition taking into account the criticism and the original intended meaning given in high-level language. The formal definitions of security properties including transparency found in Subsections 11.6.4 and 11.13.4 are the result. Only recently the full set of possible transparency variations, which are not of prime interest of this thesis, have been formally defined and shown to be achievable: Namely, *Ateniese et al.* [12] initially proposed an "invisible sanitizable signature" [12] for which *Brzuska et al.* [64] said it was "an overly strong requirement", but a positive answer was given in 2017 by *Camenisch, Derler, Krenn, Pöhls, Samelin, and Slamanig* [107] (see Appendix A publication n$^{\text{o}}$ 55).

However, many works in the state-of-the-art did not show interest in this direction and continued to focus on schemes that obtained a stronger privacy by achieving transparency[764]. Existing work even strengthened it further: The schemes from *Ahn et al.* [6] and from *Attrapadung et al.* [15] allow for *quoting* substrings and achieve both, statistical transparency and statistical unlinkability. This ambitious goal comes at the price of weakening the unforgeability property to selective unforgeability in the case of [5, 6]. Further, note that none of the mentioned schemes is any longer accountable.

While the goal of transparency is interesting, it is not essentially required in all applications[765]. Especially, transparency becomes obsolete when the fact that sanitization(s) or redaction(s) have taken place is detectable solely from obvious modifications to the message's content, i.e., context information. This happens because the cryptographic protection of transparency is solely concerned with restricting the adversaries' information gain that can be obtained through the sanitized message (m') and its derived signature (σ').[766] Hence, detecting the presence of authorized subsequent modifications from external context information is not considered a breach of the cryptographic notion of transparency.

Legally, it is necessary to provide accountability and it is beneficial to provide it even non-interactively. Thus, the thesis argues having non-interactive public accountability is preferable as it allows to increase legal certainty. As discussed in Sec. 6.4.5 it removes the need for a correct and honest Verifier to depend on other entities to make the judge's interactively-run accusation-rebuttal protocol execution to be the same as the previous ones of the Verifier. Simply because in the case of non-interactive public accountability, there is no interaction. As such the Verifier can not run into problems due to an interactively involved party not being available — or maliciously pretending to not being able to — generate repudiation-evidence.

Non-interactive Public Accountability makes the MSS become more aligned in its behaviour with a CDSS: The goal of CDSS is that a complete and in-depth verification of a signature yields:

- a verification of content, structural and referential integrity of the signed content, and
- a cryptographic proof that allows to technically counter a false repudiation of origin, i.e., provide non-repudiation of origin.

With MSS doing an authorized subsequent modification would not require an additional interaction with the Signer for the entity that sanitizes the message. However, if accountability would **not** be public and non-interactive — but only interactive — this would mean that a full verification would require such an additional interaction with the Signer: To obtain non-repudiation requires to verify the accountability,

[763] This was also mentioned in the joint work in which the notion of non-interactive public accountability was published (see Appendix A publication n$^{\text{o}}$ 11)

[764] Note, that transparency is the cryptographically stronger notion compared to privacy, i.e. transparency implies privacy (see Sec. 11.7).

[765] Here, the thesis is in line with the authors that coined the term for SSS *Ateniese et al.* who stated "[...] that strong transparency is not always better.[...] For example, if a document originally signed by some government official is later released by a certain government agency — acting as a censor — under the Freedom of Information Act, the general public would likely prefer knowing which parts of the document could have been sanitized. " [12].

[766] This follows from Analysis Result 22 based on *Brzuska et al.*'s statement saying: "As information leakage through the modified message itself can never be prevented, we only refer to information which is available through the sanitized signature." [64].

thus an interaction with the Signer is needed for accountability. This is in contrast to classic digital signatures that offer this without involvement of the Signer at the time of verification (see also Sec. 6.4 for a detailed discussion). If evidence for the origin, that can technically not be repudiated, can be generated non-interactively and publicly, then the evidence can be generated by the Verifier without the other parties (Signer nor Sanitizer) being involved and just with the knowledge of the message, the signature and the public keys of the potentially accountable parties.

12.4. Shortcoming 4: No accountability on a the fine-grained scope of blocks, only on the scope of the message

An introductory example will highlight that certain requirements cannot be fulfilled when the SSS only allows to decide which party is accountable for the entire message-signature pair.

EXAMPLE

The following example will highlight the impact of Shortcoming 4. Assume a signed medical record is allowed to be sanitized and allows certain subsequent modifications for data protection. Assume further that the SSS used to sign the data even offers the new property of non-interactive public accountability. This means that the Verifier can always detect that the medical record itself has changed. However, if more than one entry of the medical record, each represented by one block, is modifiable, the verifying party may need to know which admissible block(s) have actually undergone a sanitization. This would allow to filter potentially-sanitized from still-original information. To achieve this, the application demands more than non-interactive public accountability as introduced above, it demands accountability for *each individual* block $m[i]$ of the received message. Thus, when medical records are anonymised, e.g., to pass them to the World Health Organization (WHO), the Sanitizer is allowed to replace names (and further identifying information). Assume that for de-identification — as for instance required by US HIPAA [484] — the Sanitizer replaces the names with zeroes. In order to do this the Sanitizer must be allowed to modify the name field. A malicious Sanitizer can exchange two patients' medical records by swapping their names by sanitizing both medical records. Namely, instead of putting in a zero, the malicious Sanitizer puts in the other patient's real name. Damage is done when the swapped medical records in the hospital's database are consulted for the patients' treatment without knowing that sanitization has taken place on the name field and that it therefore must be given less trust.

The state-of-the-art accountability of transparent SSS allowed only discovering interactively with the Signer that a third party is accountable for the message that successfully verified. To make sure that authorized modified data is not becoming an attack vector, as described in the example, it requires an additional and successful interaction with the Signer before knowing if data is potentially-sanitized or original. This is a drawback, especially if the Signer can not be reached. A first mitigation would be to overcome Shortcoming 3[767] and have non-interactive public accountability (PUB). However, as sanitization is usually allowed on multiple blocks of the message knowing exactly which of these blocks are still unchanged is important: First, it allows to assign different levels of trust to information from sanitized or original blocks. Second, in a multi-sanitizer setting[768] several Sanitizers might be allowed to modifythe same block and they might have different trust ratings.

Not being able to achieve non-interactive public accountability on the level of blocks with the drawbacks it brings could hinder the applicability of RSS or SSS in many scenarios which currently deploy traditional CDSS-based authenticity protection. Even if the MSS would allow a message-level non-interactive public accountability, which is Shortcoming 3, the explicit splitting of documents into blocks as done by the MSS, might require the MSS to be more precise in its accountability property than the traditional CDSS which considers the message to be whole. Thus, this problem got listed as Shortcoming 4.

[767] Shortcoming 3: No, or no explicit public form of accountability (see Sec. 12.3).

[768] The multi-sanitizer framework was introduced by *Canard et al.* [112]. It is interesting and possible to extend the ideas of this thesis to multiple sanitizer, e.g. as discussed here Sec. 7.4.3 on page 176, but not in the scope of this thesis as mentioned in Sec. 1.4.1.

12.5. Shortcoming 5: Limited control of the structural integrity protection

This thesis takes into account the flexibility and the controllability with which the solutions allow the authorized modifications to affect the structural integrity protection, especially redaction. Many of the schemes from the body of the state-of-the-art are not allowing the full flexibility for flexibility compared to the data structure they protect would offer. To give a few examples: *Brzuska et al.* [66] protects tree-based data but allows only the redactions of leaf-nodes from the tree. Also, always all subtrees can be redacted, the Signer has no further control, e.g. the Signer can not prohibit the redaction of certain subtrees. *Ahn et al.* [5] caters for a special sub-case of redaction from an ordered list of blocks called "quoting" [5]. Quoting is not as flexible as a freely independent redaction of any list element. It resembles the functionality of "content extraction" from *Steinfeld et al.* [455] by requiring that a single and still consecutive part from the list of elements must remain in the original order — with the original content — in the redacted list. Other schemes do not explicitly consider tree-structured data, even though they mention XML as an application: In [391] existing sanitizable and redactable signature schemes are integrated into XML Signatures, extending the work done in [463]. Neither the approach of *Tan et al.*, nor the work of *Pöhls et al.* do secure trees, since neither of them caters explicitly for the structure. Of course, one solution is to drop the structural integrity protection altogether and work only on structureless sets [348].

Allowing flexibility can influence the security level, also negatively: The uncontrolled level-promotion attack against *Kundu and Bertino* [295, 296] described under Shortcoming 2 in Sec. 12.5.1.3 has highlighted that the added flexibility to redact non-leaf nodes of the tree-based integrity protected data comes at the price of a reduced structural integrity protection. This limits the integrity protection that can be offered when giving flexibility. Limited integrity protection means that some aspects of the document's integrity can not be protected by the scheme itself, e.g., only sets of blocks get protected but not an ordered list. This can lead to unwanted attack possibilities as presented in Sec. 12.5.1. However, limited flexibility means that the Signer's ability to explain his explicit consent to the authorized subsequent modifications is limited by what the scheme allows. In the following Sec. 12.5.2 highlights the applications that are made possible if structure which carries information becomes a redactable blocks in its own.

12.5.1. Problems and limits on structural protection and missing flexibility for non-leaf node redaction

In the following, different aspects are discussed to show why the following existing schemes have Shortcoming 5. When schemes offer more flexibility then extra care needs to be taken to describe and ensure the exact privacy and integrity guarantees offered. Sec. 12.5.1.3 shows that if in tree structured data the removal of non-leaf nodes is to be allowed for flexibility, like in the schemes of *Kundu and Bertino* [295, 296], then there must be extra protection against problems of limited structural integrity protection. Sec. 12.5.1.3 presents the impact of the an attack termed level-promotion attacks that reduce the structural integrity protection offered by the RSS.

Due to the attacks possible, schemes that have not been proven to fulfil Requirement 1 are not considered suitable for any application that seeks to protect the integrity against malicious subsequent manipulation, neither that of trade secrets nor that of personal data.

12.5.1.1. Limitations of *Ahn et al.* towards flexibility

The scheme introduced by *Ahn et al.* and published as [5–7] provides stronger privacy guarantees that are above the standard privacy that is used as a baseline in this thesis (see Sec. 11.6.3 on page 263 for the harmonised definition). However, this increased security comes with the following shortcomings towards its flexibility: It caters for a special sub-case of redaction called "quoting" [5]. Quoting is not as flexible as allowing to redact individual blocks from the document. However, it could be seen to resemble what *Steinfeld et al.* originally termed "content extraction" by allowing to preserve the integrity for any list of elements that still resemble consecutive elements from the original ordered list[769].

[769] This thesis uses the term ordered list, to explicitly indicate that the the order shall be preserved by the structural protection.

12.5.1.2. Limitations of *Camacho and Hevia* towards integrity

Also recently, *Camacho and Hevia* have shown how to build more efficient transitive signatures for directed trees [103]. Their scheme allows for removal of non-leaf nodes in a tree. They are based on an idea from *Rivest* and *Micali* [339]. However, the work from *Micali and Rivest* [339] focuses on how to authenticate single edges within a signed tree T. As such, *Camacho and Hevia* did also not consider the invisibility of redaction.

12.5.1.3. Problems of schemes by *Kundu et al.* towards integrity due to level-promotion attack

The schemes by *Kundu et al.* [295, 296] are the only existing RSS for trees able to redact non-leaf nodes. However, previously several attacks on the schemes by *Kundu et al.*were presented. Namely on the constructions from [296] and from [295]. The attacks showed that also in its latest revised form [295] the scheme does not always preserve the structural integrity of the tree-based document (see discussion on Shortcoming 2 in Sec. 12.2.1). *Kundu et al.* [295, 296] offer limited structural integrity protection: Those schemes, while for tree-structured data, are only protecting the ancestor-of relation when it comes to structural integrity protection. Hence, these schemes do not suffice if a document's semantic meaning is changing if the the level of a node inside the tree or the direct father-child relationship of nodes carries information that must be protected against unauthorized modifications.

EXAMPLE

To highlight the negative impact, consider for example that the structure to be signed is a hierarchical structure of treatments inside a medical database. Treatments themselves can consists of treatments, this might be codified in the data structure, e.g. as an XML schema. Assume the treatment 'chemotherapy' consists of two sub-treatments: 'cancer drugs' and an additional 'prophylactic drugs' to avoid infections. Assume that this is codified into the tree's structure, i.e. a structured electronic health document [156]. If the 'cancer drugs' and the 'chemotherapy' node are both individually redacted, the sub-treatment node for the 'prophylactic drugs' is left alone in with no treatment above. That this treatment was now promoted is not detected syntactically in this case, i.e., the level-promotion is not detected by a failing XML schema validation.

12.5.2. Motivation for structure becoming individually redactable

Still, whether or not a certain limitation in the structural integrity protection is acceptable clearly depends on the exact application. To enable full flexibility and retain the general ability to redact intermediate nodes of a tree, it is necessary to manipulate the structure in a Signer- controlled way. In general, this thesis identified in Sec. 5.5.1 that it is the application that understands what modifications shall be allowed. This got codified into Analysis Result 8[770].

Still, it can be the structure itself which does carry the information. In the following this thesis states three examples to highlight the gains of being able to redact structure individually. Hence, a flexible scheme shall offer a fine-grained control and the Signer shall explicitly denote the places where a violation of structural integrity protection is consented.

EXAMPLE

As a first example, imagine a tree of employees' names that is in a hierarchical tree where the companies structure is encoded into the tree's level. Then explicitly allowing level-promotion can be used to redact hierarchies within a company.

EXAMPLE

In the second example, *Alice*, *Bob* and *Carol* are students that took the same exam. The university administration generates a signed list containing for each student the name followed by the student's grade in the exam. Let there be ℓ students' names and grades, e.g. the list has 2ℓ elements. Imagine the list being ordered by grades, i.e., the student with the best grade is at the beginning. Such an ordered list is depicted in Fig. 67. Assume the university is using a secure redactable signature scheme (RSS) to generate one single signature over the list.

[770] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

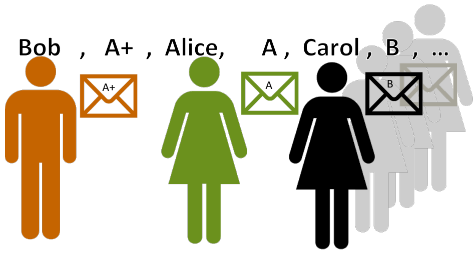


Figure 67. Original list of student's name and grade, ordered by grade

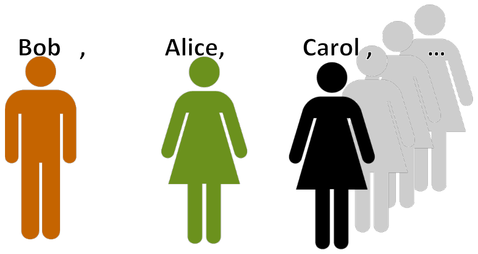


Figure 68. Redacting all list elements containing the grade

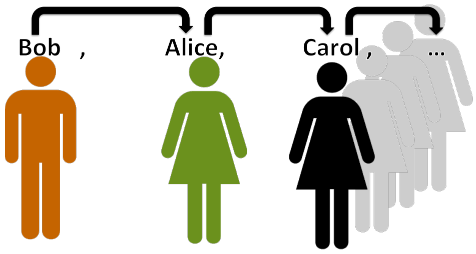


Figure 69. Not redacted list elements remain ordered by grade

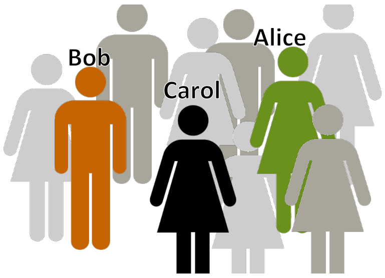


Figure 70. Goal: Remove all list elements containing grades and remove the order among elements

If signed using an RSS each block $m[i]$ holds information about either a student's grade or a student's name, while the ordering carries the information "better-than". Imagine to redact that list such that it is a signed set of all the students' names who took part in the exam. Further it is desired that the redaction removes *all* information about grades from this list. In its most flexible case an RSS shall allow anyone to *redact* an element from this list. Consecutive redaction then deletes all grades, i.e. in this example all $m[i] \leftarrow$ ▮, where i is even. When a transparent RSS, e.g. [12, 64, 66], would be used this also removes the trace that an element of the list was removed. In particular, ▮ would not be visible. The result of a transparent redaction is depicted in Fig. 68.

However, the original order still remains even though the information used for generating the order has been successfully redacted (see Fig. 69). This invades privacy. Hence, one also desires to redact the ordering among the remaining elements of the list to finally remove all the information relating to the students' grades (see Fig. 70).

EXAMPLE

The third example is taking a business point of view: The sales department provides a monthly overview over all invoices that where generated to an external auditor. However, the order in which a company sends out its invoices may leak some business-critical information. Further imagine that the list must be ordered in a monthly manner, i.e., the invoices need to be grouped by month, while the internal ordering must not be made public and can remain confidential to protect trade-secrets.

There are many other application scenarios, e.g. to explicitly sign partially ordered sets, which is useful for information flow models [430]. Removing structure, first requires to identify the structure as information carrier and then to treat it accordingly. This has been done in the course of this thesis and a construction named $struct\mathcal{RSS}$ is presented in Sec. 14.9 on page 399.

12.6. Shortcoming 6: No explicit dual operation to redact, i.e., no explicit 'merge' of two linkable modified documents

The property of unlinkability has been defined for SSS in Sec. 11.6.5. However, if a scheme is not unlinkable and a party is in possession of two differently sanitized versions that were obtained by valid sanitizations from the same source, or are two stages in a consecutive sanitization it might be reasonable to assume that this party can always (or at least with overwhelming probability) link those 'variants'. Knowing if a block was sanitized or not, i.e. offer block-level accountability, would allow to restore as many original contents of the original message as contained in the combined set of two (or more) linkable 'variants' of the same document. This inverse operation is best explained for redactable signatures and was termed **merge**[771]. The operation of merge works as follows: Given two message-signature pairs derived only by authorized sanitizations from the same signature over the same message, one can combine them into a single *merged*, set-signature pair $(\mathcal{S}, \sigma)$.

Note that in the field of RSSs, all existing provably *private* constructions only consider how to redact elements. The opposite — reinstating previously redacted elements, i.e., merging signatures — in a controlled way has neither been formalised nor have security models been properly discussed. Notions of mergeability are initially given by *Merkle* for hash-trees [338], but these are not private in the context of RSSs. The closest existing works mentioning merging in this context are [273, 314].

EXAMPLE

Example: Consider a database with records containing information about a human's DNA that are associated with personal data, e.g. the person's name. While the personal data part might be quite small, the medical data is often very large and it would therefore be nice to remove unnecessary information before sharing. Additionally, maybe only certain excerpts of the full genome are of interest for a study or a diagnosis. Still, integrity is of essence to ensure that the sequences analysed are genuine and original. Figure 71 depicts the creation of variants by redaction and the re-combination by merging. For the example it is assumed that each field (`id`, `Name`) can be redacted and that each single entry for one of the four nucleobases can be redacted individually.

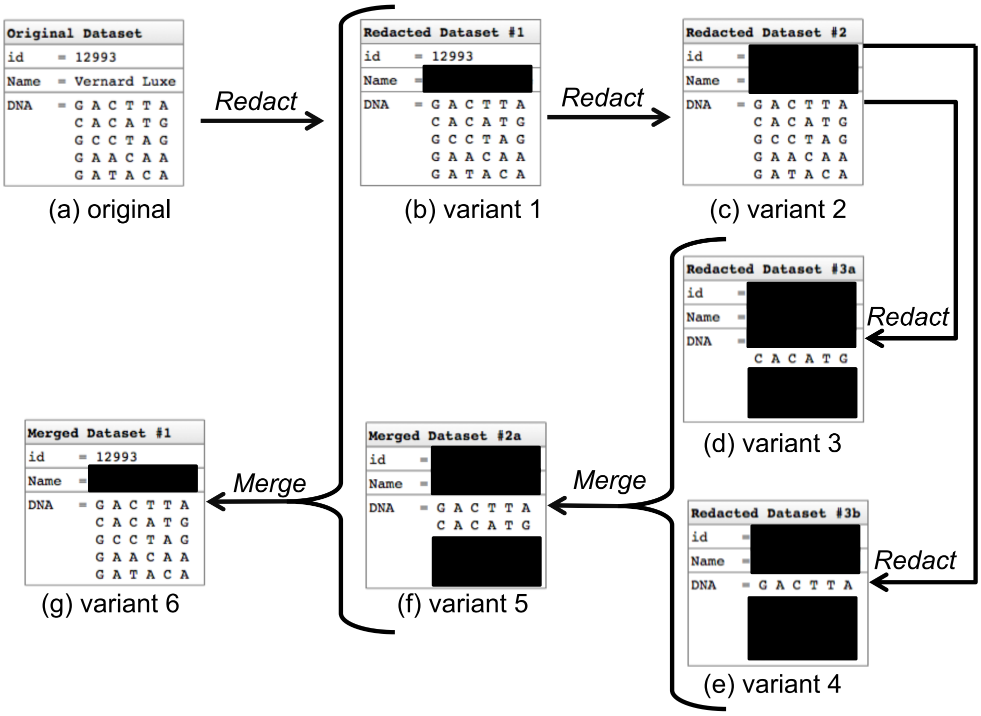


Figure 71. The operation 'merge' is an explicit inverse of the redact operation; merging allows to generate validly signed variants from any two valid, but modified variants if and only if they have a common ancestor (original or redacted variant)

Moreover the analysis of existing RSS found that schemes like the one from *Brzuska et al.* [66] or *Miyazaki et al.* [348] already offer this functionality implicitly. In detail, assume that all signatures share a common tag τ. An adversary $\mathcal{A}$ can break the state-of-the-art unforgeability [66] of an RSS in the following way: $\mathcal{A}$ queries its signing oracle with a set $\{A\}$, receiving a signature σ_A. Afterward, $\mathcal{A}$ requests the signer to update $\{A\}$ to $\{A, B\}$, receiving a signature $\sigma_{A,B}$. Additionally, $\mathcal{A}$

[771] The results presented here for RSS have been published as joint work with *K. Samelin* at the ACNS conference in 2014 [387] (see Appendix A publication n° 20).

requests a second update of $\{A\}$ to $\{A,C\}$, receiving $\sigma_{A,C}$. $\mathcal{A}$ can then "merge" $(\{A,B\},\sigma_{A,B})$ and $(\{A,C\},\sigma_{A,C})$ to a new verifying signature $(\{A,B,C\},\sigma^*)$. This set-signature pair is considered a forgery in existing models, while this may be a wanted behaviour, e.g., if in medical records new diseases are appended by two different medical doctors.

Note, in the above example a new oracle, namely the update-oracle, has been introduced. It models that an adversary can request the Signer to dynamically update an already signed set. In a nutshell, dynamic updates allow the signer to add new elements to existing signatures. This captures the ideas given in [40, 281]. Existing provably secure[772] constructions offer the possibility of such dynamic updates.

However, even if the adversary cannot request updates of his own choosing, it can merge them. This is of particular interest, as this very subtle possibility undermines existing schemes' unforgeability in current security models. The merged signature might fall into what was considered a forgery, according to existing models, once the Signer performs two different updates on a signed set. Hence, to overcome this shortcoming either dynamic updates must be completely prohibited, or the existing security model must be altered.

For certain applications dynamic updates might have their merits and enable many practical applications. A Signer can add new elements without the need to re-sign everything. This is useful in situations were it is too costly to re-sign a database completely every time a new entry is added. In other words, updates may become less expensive than a complete re-sign. Therefore, this thesis did not find a suppression of dynamic updates a viable solution and lists this as Shortcoming 6. Note, that the given example is tailored for sets, while some schemes address lists and trees. However, the aforementioned possibility is not limited to sets, but works on all the schemes aforementioned with minor adjustments. An exception are schemes which offer context-hiding (and their variants). This strong privacy property discourages any discussions about dynamic updates, as an updated signature cannot be linked against an "old" one.

This thesis introduces a new framework in Sec. 14.2 called MRSS which offers two additional algorithms Update and Merge. Thus the merge operation gets explicitly defined, while in existing work it is possible but it only *implicitly* defined. This also allows the security model to be adjusted accordingly.

As a side effect, the ability to explicitly check for a successful mergeability creates and algorithm to identify *linkable* variants of the same document: once two signed sets are available, their signatures only merge, if they are linkable. This enhances the usability as in some applications an adversary must not be able to generate clones of a signed set by gradually redacting different elements.

EXAMPLE

As an example assume that `Name` consists of several columns (`FirstName`, `MiddleName`, `FamilyName`). Then, it might be a good idea to prohibit that the DNA record of 'Rose' 'Fitzgerald' 'Kennedy' can be duplicated without detection into several sets by redaction of some of the elements, e.g., pretending two signed DNA records exist: One for a member of the 'Kennedy' family and another one for a woman with the first name of 'Rose'.

[772] As defined in Sec. 2.3.1.

13 — Proposed private SSS with an increased probative value

Overview of Chapter 13

Chapter 13 proposes three cryptographic properties that go beyond the state of the art. The properties are each accompanied by a proposed SSS construction, which is proven to achieve the desired proposed property. All three proposed SSS schemes, as well as all properties, got published to disseminate the results. An overview is shown in Tab. 11.

Property	Scheme's name	Section	Publication(s)
Non-interactive public accountability with privacy (13.1)	$pubacc\mathcal{SSS}$	13.4	[68]
Accountability, both transparent and publicly non-interactive, on the scope of individual or groups of block(s) (13.2)	$blockacc\mathcal{SSS}$	13.6	[132, 133]
Strengthened unlinkability (13.3)	$unlinkable\mathcal{SSS}$	13.8	[69]

Table 11. Overview of new SSS properties and the three new SSS schemes presented in this thesis

$pubacc\mathcal{SSS}$ is the answer to the call for a scheme with non-interactive public accountability. It achieves non-interactive public accountability, privacy, immutability, unforgeability and drops transparency. Thus, it achieves the same level of detectability of a CDSS, i.e., $pubacc\mathcal{SSS}$ achieves $\text{ACA}-\geq 1\text{CD}-\text{PUB}$ integrity (see Requirement 3[773] in Sec. 7.3 and Requirement 4[774] in Sec. 7.4). To add additional usefulness to the accountability, $pubacc\mathcal{SSS}$ offers fine-grained information on the level of blocks (see Sec. 6.5). With block-level non-interactive public accountability the Verifier is equipped with an algorithm to identify which blocks are still original (unchanged) and which have been subsequently modified with authorization of the Signer. Of course $pubacc\mathcal{SSS}$ detects any unauthorized modification and achieves privacy according to *Brzuska et al.* This versatile scheme was also implemented and integrated into a supply chain system: Sec. 18.1 presents some details of this full system integration of an SSS. Further, $pubacc\mathcal{SSS}$ can run on a smart card, i.e., on a QSCD: Sec. 13.10.2 presents a proof-of-concept implementation.

$blockacc\mathcal{SSS}$ achieves non-interactive public or transparent accountability on the level of groups of blocks, and alongside the necessary baseline properties of privacy, immutability, unforgeability. It adds more flexibility for the Signer as it allows to combine blocks into groups for the accountability.

$unlinkable\mathcal{SSS}$ provides the construction for an SSS that achieves unlinkability that is robust against malicious or buggy signers.

Finally, a proof of concept implementation shows that $pubacc\mathcal{SSS}$ can run on a smartcard as a QSCD. The presentation of this implementation in Sec. 13.10.2 concludes Chapter 13. The five proposed RSS are presented in Chapter 14.

[773] Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification) (see Sec. 7.3).

[774] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

13.1. Proposed SSS property: Non-interactive public accountability (PUB)

The proposed property of non-interactive public accountability[775] is the dual of the property of transparency. It is the explicitly public form of accountability. Accountability was defined in general for digital signature schemes in Definition 23 (see Sec. 2.10.2). In general and also for MSS, accountability is independent of transparency [64]. However, the level of detection might be limited to interactive accountability for transparent schemes. Moreover, note that non-interactive public accountability is not obtained by a negation of transparency (see Theorem 5):

$$\neg\,\text{transparency} \not\Rightarrow \text{non-interactive public accountability}\,.$$

The details are shown soon in Sec. 13.1.3. Examples have already been given in Sec. 3.10.2.2. The property of non-interactive public accountability requires the integrity protection to allow a third party to decide whether a Sanitizer's secret key was involved in the signature generation or if only the Signer's secret key was involved even in the presence of a malicious or non-cooperative Signer/Sanitizer. In other words, only with the knowledge of the validly signed message-signature pair and the public keys, the decision about the respective secret key's involvement can be made. Thus, by linking the public key to the corresponding entity it is possible to publicly and non-interactively determine the origin of the signature.

13.1.1. Shortcoming 3 addressed by non-interactive public accountability

The new property for SSS of *non-interactive public* accountability embodies that the assignment of the party that is accountable can be proven non-interactively and just with the knowledge of public keys and the message-signature pair. As found in Analysis Result 12[776] accountability "[...] ensures that the actions of an entity may be traced uniquely to the entity." [239, 263]. It directly addresses the Requirement 3[777] on accountability as described in Sec. 7.3. Further, if the accountability can be established non-interactively, then also the Verifier is enabled to check the Signer's proof of consent to a reduced integrity[778] in a non-interactive and public way. Thereby, non-interactive public accountability also fulfils Requirement 2.3[779]. Thereby it addresses the Shortcoming 3[780] that was described and motivated in Sec. 12.3. This public form is especially helpful as it eases to argue that non-interactively publicly accountable MSS offer an increased probative value as their behaviour. The reason is that due to non-interactive public accountability the MSS can provide the same non-repudiation and accountability services that are provided by standard signatures (see Sec. 12.3 and Sec. 6.4.5).

13.1.2. Definition of non-interactive public accountability (PUB) on the block- and message-level

For the message-level this thesis proposes to define the notion of *non-interactive public accountability* as follows:

[775] The new property non-interactive public accountability and the results given in this section have in part already been published at EuroPKI 2012 as joint work with *C. Brzuska* and *K. Samelin* [68] (see Appendix A publication n° 11). The thesis already contains some minor improvements made compared to the originally published paper. The updates are: an altered block accountability game to address mix-and-match attacks; $m[\mathsf{fix}]$ needs to part of the signature processing algorithms in the second construction; order of parameters for consistency.

[776] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

[777] Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification) (see Sec. 7.3).

[778] The integrity is 'reduced' as it is most probably less than the usually expected $\text{NCA}-\geq 1\text{CD}$ integrity that CDSS are expected to offer.

[779] Requirement 2.3 (Mechanism SHOULD enable Verifier to verify the proof of consent non-interactively and publicly)

[780] Shortcoming 3: No, or no explicit public form of accountability (see Sec. 12.3).

Definition 179 : Non-Interactive Public Accountability (PUB)

*A sanitizable signature scheme satisfies **non-interactive public accountability** (PUB), if and only if for a valid message-signature pair (m, σ), a third party can correctly decide whether (m, σ) originates from the Signer or from the Sanitizer without requiring an interaction with the Signer, the Sanitizer or any other party with access to secret keys, i.e. just from using public knowledge $(m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san})$.*

In contrast to the game of interactive non-public accountability (see Definition 160 on page 271), the algorithm Judge must decide correctly even on an empty proof π. This would no longer require to execute Proof, which needed a secret key. This captures the required form of public accountability. Formally, this results in the following definition for the message-level:

Definition 180 : Message-level Non-Interactive Public Accountability (PUB) for SSS

*An SSS is **non-interactive publicly accountable**, if Proof $= \perp$, and, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment Pub-Accountability$_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 72 returns 1 is negligible (as a function of λ).*

```
Experiment $\mathsf{Pub\text{-}Accountability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
    $(\mathsf{pk}_{\mathsf{sig}}, \mathsf{sk}_{\mathsf{sig}}) \leftarrow \mathsf{KGen}_{\mathsf{sig}}(1^\lambda)$
    $(\mathsf{pk}_{\mathsf{san}}, \mathsf{sk}_{\mathsf{san}}) \leftarrow \mathsf{KGen}_{\mathsf{san}}(1^\lambda)$
    $(\mathsf{pk}^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathsf{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathsf{san}})}(1^\lambda, \mathsf{pk}_{\mathsf{sig}}, \mathsf{pk}_{\mathsf{san}})$
        for $i = 1, \ldots, q$ let $(m_i, \mathsf{ADM}_i, \mathsf{pk}_{\mathsf{san},i})$ and $\sigma_i$
        index the adaptive queries and answers to and from the oracle Sign
        return 1 if
            $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathsf{sig}}, \mathsf{pk}^*) = \mathtt{true}$, and
            $\forall i \in \{1, \ldots, q\} : (\mathsf{pk}^*, m^*, \mathsf{ADM}^*) \neq (\mathsf{pk}_{\mathsf{san},i}, m_i, \mathsf{ADM}_i)$ and
            $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathsf{sig}}, \mathsf{pk}^*, \perp) = \mathtt{Sig}$
        for $j = 1, \ldots, q'$ let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk}_{\mathsf{sig},j})$ and $(m_j', \sigma_j')$
        index the adaptive queries and answers to and from the oracle Sanit
        return 1, if
            $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathsf{san}}) = \mathtt{true}$, and
            $\forall j \in \{1, \ldots, q'\} : (\mathsf{pk}^*, m^*, \mathsf{ADM}^*) \neq (\mathsf{pk}_{\mathsf{sig},j}, m_j', \mathsf{ADM}_j)$ and
            $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathsf{san}}, \perp) = \mathtt{San}$
```

Figure 72. Message-level Non-Interactive Public Accountability (PUB) Experiment for SSS

Like with the interactive accountability notions, an adversary, e.g., the sanitizer or the signer, has to be able to make the algorithm Judge decide incorrectly in their favour. This results in the requirement that the adversary must either provide a message-signature pair (m^*, σ^*) for which Judge decides San even if it was never been output by Sanit or decide Sig even if it was never been output by Sign.

A strengthened non-interactive public accountability notion was given by *Krenn et al.* [294] in 2015. It further includes the signatures as part of the 'freshness' check. In the same order and notation as Definition 180[781], this amendment would result in the following definition:

Definition 181 : Strong Message-level Non-Interactive Public Accountability (PUB) for SSS following *Krenn et al.*

*An SSS is **strongly non-interactive publicly accountable**, if Proof $= \perp$, and, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment Strong-Pub-Accountability$_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 73 returns 1 is negligible (as a function of λ).*

[781] This is the notion of non-interactive public accountability (PUB) as it was published in the joint work [68] (see Appendix A publication n° 11).

```
**Experiment** $\mathsf{Strong\text{-}Pub\text{-}Accountability}^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
    $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
    $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
    $(\mathsf{pk}^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}})}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
        for $i = 1, \ldots, q$ let $(m_i, \mathsf{ADM}_i, \mathsf{pk}_{\mathrm{san},i})$ and $\sigma_i$
        index the adaptive queries and answers to and from the oracle Sign
        return 1, if
            $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*) = \mathtt{true}$, and
            $\forall i \in \{1, \ldots, q\} : (\mathsf{pk}^*, m^*, \mathsf{ADM}^*, \sigma^*) \neq (\mathsf{pk}_{\mathrm{san},i}, m_i, \mathsf{ADM}_i, \sigma_i)$ and
            $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*, \perp) = \mathtt{Sig}$
        for $j = 1, \ldots, q'$ let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk}_{\mathrm{sig},j})$ and $(m'_j, \sigma'_j)$
        index the adaptive queries and answers to and from the oracle Sanit
        return 1, if
            $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}) = \mathtt{true}$, and
            $\forall j \in \{1, \ldots, q'\} : (\mathsf{pk}^*, m^*, \mathsf{ADM}^*, \sigma^*) \neq (\mathsf{pk}_{\mathrm{sig}_{,j}}, m'_j, \mathsf{ADM}_j, \sigma_j)$ and
            $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}, \perp) = \mathtt{San}$
```

Figure 73. Strong Message-level Non-Interactive Public Accountability (PUB) Experiment for SSS following *Krenn et al.* [294]

Note that *Krenn et al.* [294] checked that the construction $pubacc\mathcal{SSS}$ (see Sec. 13.4.3 on page 331) [68] already achieves the stronger notion even though this was not intended. The construction of $pubacc\mathcal{SSS}$ and the underlying ideas were published in 2012[782]. Namely, *Krenn et al.* [294] state the following: "The construction given in [68] achieves non-interactive public accountability. Due to the determinism and uniqueness of the signature schemes deployed, their scheme already fulfills our strong definitions without any modifications."[783] [784] [294].

For the block-level non-interactive public accountability (PUB), this thesis defines an additional algorithm denoted blkJudge. Analogue to the goal of the algorithm Judge which allows the Verifier to publicly verify accountability for the message as a whole towards the Signer or the Sanitizer, blkJudge indicates which party is accountable for the i^{th} block. Most notably, compared to the input of a message-level algorithm like Judge, blkJudge additionally demands as an input the block's index i and then returns San or Sig. With the additional algorithm the property of block-level non-interactive public accountability (PUB) can be described — in the non strengthened version — as follows:

Definition 182 : Block-level Non-Interactive Public Accountability

*A sanitizable signature scheme SSS together with an algorithm blkJudge is **publicly accountable on the level of blocks**, if Proof $= \perp$ and if for any efficient algorithm $\mathcal{A}$ the probability that the experiment Block-Pub-Accountability$^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$ given in Fig. 74 returns* 1 *is negligible (as a function of λ).*

The basic idea is that an adversary, i.e., the Sanitizer or the Signer, wins if it is able to make the block-level judge algorithm blkJudge decide wrongly on any validly signed block given an empty proof π.

[782] The results were published jointly with *C. Brzuska* and *K. Samelin* at EuroPKI 2012 [68] (see Appendix A publication n° 11).

[783] The American English spelling from original work is retained for terms and quotes.

[784] Original's endnote has been adjusted to correctly point to the same entry found in the bibliography of this thesis.

```
Experiment $\mathsf{Block\text{-}Pub\text{-}Accountability}^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
    $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^{\lambda})$
    $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^{\lambda})$
    $\mathcal{Q}_{\mathrm{sig}} \leftarrow \emptyset$
    $\mathcal{Q}_{\mathrm{san}} \leftarrow \emptyset$
    $(\mathsf{pk}^*, m^*, \sigma^*, q^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(\cdot,\mathsf{sk}_{\mathrm{sig}},\cdot,\cdot),\mathsf{Sanit}(\cdot,\cdot,\cdot,\cdot,\mathsf{sk}_{\mathrm{san}})}$
        Let $(m_i, \mathsf{ADM}_i, \mathsf{pk}_{\mathrm{san,i}})$ and $\sigma_i$ for $i = 1, 2, \ldots, m$
        be the queries and answers to and from oracle Sign.
        For every query to Sign, let
        $\mathcal{Q}_{\mathrm{sig}} \leftarrow \mathcal{Q}_{\mathrm{sig}} \cup \{(\mathsf{pk}_{\mathrm{san,i}}, \mathsf{ADM}_i, m[\mathsf{fix}]_i, (m_i[k], k))\}$ for all $k \in \{1, 2, \ldots, \ell_i\}$
        return 1, if
            $\mathsf{Verify}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*) = \mathtt{true}$, and
            $\mathsf{blkJudge}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*, \perp, q^*) = \mathtt{Sig}$ and
            $(\mathsf{pk}^*, \mathsf{ADM}^*, m[\mathsf{fix}]^*, (m^*[q^*], q^*)) \notin \mathcal{Q}_{\mathrm{sig}}$
        Let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk}_{\mathrm{sig,j}})$, $(m'_j, \sigma'_j)$ for $j = 1, 2, \ldots, m'$
        be the queries and answers to and from oracle Sanit.
        For every query to Sanit, let
        $\mathcal{Q}_{\mathrm{san}} \leftarrow \mathcal{Q}_{\mathrm{san}} \cup \{(\mathsf{pk}_{\mathrm{sig,j}}, \mathsf{ADM}_j, m[\mathsf{fix}]_j, (m_j[l], l))\}$ for all $l \in \{1, 2, \ldots, \ell_j\}$
        return 1, if
            $\mathsf{Verify}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}) = \mathtt{true}$, and
            $\mathsf{blkJudge}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}, \perp, q^*) = \mathtt{San}$, and
            $(\mathsf{pk}^*, \mathsf{ADM}^*, m[\mathsf{fix}]^*, (m^*[q^*], q^*)) \notin \mathcal{Q}_{\mathrm{san}}$
    return 0.
```

Figure 74. Block-level Non-Interactive Public Accountability (PUB) Experiment for SSS

13.1.3. Relation between non-interactive public accountability and transparency

By inspection of the definitions, one obtains the following theorem:

Theorem 4 : Non-interactive Public Accountability $\Rightarrow$ Accountability

> *Let SSS = (KGen$_{sig}$, KGen$_{san}$, Sign, Sanit, Verify, Proof, Judge) with Proof = $\perp$, then SSS is accountable, if SSS is non-interactive publicly accountable.*

Proof 4

> *In an accountable SSS, invocation of Judge is public, but needs the proof π generated using secret key sk$_{sig}$ as input to Proof. A correctly working Judge gives the correct answer with overwhelming probability on input of a correct proof π. In a non-interactive public accountability scheme, the invocation of Proof outputs $\perp$. Nonetheless, the public Judge gives the correct answer with overwhelming probability on the input of $\pi = \perp$. If and only if the non-interactive public accountability scheme's Judge is correct, the scheme is therefore still accountable.* □

The property of transparency does not prohibit all forms of accountability (see Definition 157 and Definition 158 in Sec. 11.6.4). However, it requires the Verifier to involve another party that knows secrets, to run the Judge algorithm. Thus, it is by definition mutually exclusive to the non-interactive public accountability as defined in Definition 112. Note, transparency is not a mandatory property for a secure, and especially private MSS (see previous discussion in Sec. 3.10.2).

On first sight, one might expect that in the absence of transparency it is easy to determine which party issued a message-signature pair. *Agrawal et al.* adequately described the goal as "[t]he verifier knows which part of the document is sanitized" [4], but wrongly termed this property "No transparency" [4]. The following counter example shows that this line of reasoning on negating transparency, also due to the probabilistic definition, is incorrect:
When one is able to correctly detect $\frac{3}{4}$ of all Sanitizer involvements, the scheme is neither accountable, nor transparent. Moreover, a lack of transparency does not guarantee security against malicious Signers/Sanitizers. Hence, the mechanism must allow a third party to decide whether a Sanitizer's secret

key was involved in the signature generation or if only the Signer's secret key was involved even in the presence of malicious Signers/Sanitizers. The key's involvement can then be linked to the corresponding parties and determine the origin of the signature. Following the existing definition of transparency — for example transparency for SSS from Definition 158 — Theorem 5 shows that it is indeed *in*sufficient to negate the property of transparency to gain the new property.

Theorem 5 : A Scheme can at the same time offer $\neg$ transparency and $\neg$ non-interactive public accountability

There exists an SSS which achieves neither transparency nor non-interactive public accountability. Hence,

$$\neg \textit{ transparency} \not\Rightarrow \textit{ non-interactive public accountability},$$

$$\neg \textit{ non-interactive public accountability} \not\Rightarrow \textit{ transparency}.$$

Proof 5

The proof of the theorem is given through the construction of a concrete scheme. Let SSS$_1$ be a transparent SSS with transparency as defined in Definition 158. Then one alters the algorithms as follows: when signing, the Signer appends a 0 to the signature. The Sanitizer, when sanitizing, appends a 1 to the signature. Clearly, signatures from honest Signers can be easily distinguished from signatures of honest Sanitizers. This clearly breaks transparency for honest Signers and Sanitizers. However, a malicious Sanitizer could append 0 instead of 1. The additional bit is not otherwise protected. Respectively, a malicious Signer could append a 1 instead of 0. Thus, when assuming the presence of such malicious behaviour, a third party fails to correctly determine the real origin by just the inspection of the appended bit and without executing Proof and Judge.

To separate the two notions even for honest Signers and Sanitizers, one can alter the above example in such a way that the Sanitizer appends a random bit. Now, determining the correct origin of a Sanitizer's signature succeeds with probability $\frac{1}{2}$, which still breaks transparency, that requires it to be negligibly close to 0. Also, the probability of $\frac{1}{2}$ does not satisfy public accountability, that requires it to be negligibly close to 1. □

Already from the definition of transparency and non-interactive public accountability it follows that they are mutually exclusive. Hence, this thesis captures the following two relations as definitions:

Definition 183 : Non-Interactive Public Accountability $\Rightarrow$ $\neg$ Transparency

A non-interactive publicly accountable sanitizable signature scheme is not transparent.

Definition 184 : Transparency $\Rightarrow$ $\neg$ Non-Interactive Public Accountability

A transparent sanitizable signature scheme is not non-interactive publicly accountable.

Finally, all the relations discussed before are depicted in Fig. 75.

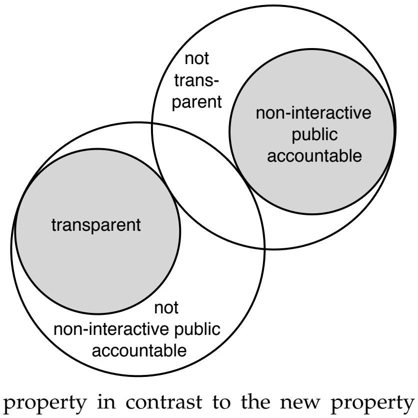


Figure 75. Transparency property in contrast to the new property of non-interactive public accountability

13.1.4. Relation between non-interactive public accountability and privacy

Transparency, the stronger privacy notion, is being negated by non-interactive public accountability. Still, transparency is stronger than standard privacy: *Brzuska et al.* showed transparency $\Rightarrow$ privacy [64]. Thus, to uphold privacy, the negation of transparency must be achieved without also negating privacy. Assume a non-interactively publicly accountable, i.e., non transparent, SSS and the privacy game as defined in Definition 156 (see Sec. 11.6.3). As the first message the adversary would input $m_0 = A||B$ with $\mathsf{ADM}_0 = 1$ together with an empty modification instruction $\mathsf{MOD}_0 = \varnothing$. As the second message the adversary would input $m_1 = A||C$ with $\mathsf{ADM}_1 = 1$ together with a modification instruction to change $m_1[1] = C$ into 'B', i.e., $\mathsf{MOD}_1 = (B, 1)$. The oracle would start to sanitize picking randomly an input message, as both modification instructions are within the limits of the policy and both result in the same output, i.e., $m' = A||B$. A naive technical implementation of a sanitization might not do anything if instructed to do nothing by an empty MOD. However, in a scheme with non-interactive public accountability both messages must be attributed to the Sanitizer, otherwise the adversary could know that a message m' that the Signer is accountable for would have been produced from the input m_0. This would allow the adversary to break privacy. This thesis achieved to uphold privacy by making sure that an empty modification instruction will still 'touch' the message or block, as defined in Definition 57. Note that this has been incorporated into the notion of a sanitization in Sec. 3.4.3 and when defining the modification instruction in Sec. 3.5. This is further observable in the $pubacc\mathcal{SSS}$, in the algorithmic description touching the message' is discussed in Sec. 13.4.4, as well as in the cryptographic construction presented later in Sec. 13.5 on page 332. For block- and group-level non-interactive public accountability an adversary can use the additional functionality to break standard privacy. Details are found in Sec. 13.4.1. Due to the fact that this adversarial behaviour can not be externally enforced a scheme offering block-level non-interactive public accountability will require a minor adjustment of the standard privacy definition to explicitly remove the information gained from the public and non-interactive detection of sanitized block as shown in Fig. 84 on page 330. The $pubacc\mathcal{SSS}$ construction is private and offers public non-interactive accountability of the Signer and the Sanitizer, as proven in Sec. 13.5.3 on page 336 for both message-level and block-level.

13.1.5. Relation between block-level and message-level accountability properties

The newly introduced block-level properties for accountability can be set into relation to their corresponding message-level properties. This has been published as a technical report [428] (see Appendix A publication n$^{\text{o}}$ 27).

For example the accountability properties already known for the scope of the whole message can be constructed from the more fine-grained block-level properties as follows:

Definition 185 : Message-level Accountability from Block-Level

> *A Sanitizer is accountable for a complete message-signature pair (m, σ), if and only if the SSS is block-level accountable and there exists at least one block of m, for which blkJudge identifies the Sanitizer as accountable.*
> *Vice versa, a Signer is accountable for the complete message-signature pair (m, σ), if and only if the SSS is block-level accountable and blkJudge identifies the Signer to be accountable for all blocks of m.*

This will model the expected behaviour for SSS, as originally defined by *Brzuska* et al. [64]. Note, Sec. 14.1.1 discusses that this becomes a bit counter intuitive for RSS, but still correctly reflects the legal accountability understanding.

Theorem 6 : Block-level Signer-Accountability $\Rightarrow$ Signer Accountability

> *Every SSS which is block-level Signer-accountable is also Signer accountable.*

Proof 6

> *Assume towards contradiction, that there exists a block-level Signer accountable scheme, which is not Signer-accountable on the message-level. Assume Signer-accountable on the message-level is built upon the block-level as described in Definition 185. Then there exists a tuple (m^*, σ^*, π^*) for which the genuine message-level algorithm Judge*

decides wrongly, i.e., Judge(m^*, σ^*, π^*) *outputs* `San` *even-though the message is still untouched. This implies that there exists one block $m^*[x]$ that has not been generated by the* Sanitizer*, which must be detected by the* blkJudge *to be block-level* Signer*-accountable, i.e.,* blkJudge$(m^*, \sigma^*, \pi^*, x)$ *outputs* `Sig`*. The contradiction follows from the emulation of message-level accountability from block-level as described in Definition 185.* □

In an analogue way the accountability of the Sanitizer for a message follows from block-level Sanitizer-Accountability as follows:

Theorem 7 : Block-level Sanitizer-Accountability ⇒ Sanitizer-Accountability

Every SSS *which is block-level* Sanitizer*-accountable is also* Sanitizer *accountable.*

Proof 7

Assume towards contradiction, that there exists a block-level Sanitizer *accountable scheme, which is not* Sanitizer*-accountable on the message-level. Assume* Sanitizer*-accountable on the message-level is built upon the block-level as described in Definition 185. Then there exists a tuple (m^*, σ^*, π^*) for which the genuine algorithm* Judge *decides wrongly, i.e.,* Judge(m^*, σ^*, π^*) *outputs* `Sig` *even-though the message has actually been touched by the* Sanitizer*. This implies that there exists one block $m^*[i]$ that has been generated by the* Sanitizer*. However, this block will be detected by* blkJudge *as the scheme is block-level* Sanitizer*-accountable. The contradiction follows from the emulation of message-level accountability from block-level as described in Definition 185.* □

Definition 186 : Block-level Signer-Accountability ∧ Block-level Sanitizer-Accountability ⇒ Block-level Accountability

A sanitizable signature scheme SSS *is* ***block-level accountable****, if it is block-level* Signer*-accountable and block-level* Sanitizer*-accountable.*

In the following the relation between non-interactive public accountability are intuitive and are stated for completeness but without a formal proof.

Theorem 8 : Block-level Non-Interactive Public Accountability ⇒ Public Accountability

Every block-level non-interactive publicly accountable sanitizable signature scheme is also non-interactive publicly accountable.

Proof 8 : Sketch

Following the intended simulation described in Definition 185 the algorithm for message-level non-interactive publicly accountability Judge_{Msg} *can be simulated by only access to block-level algorithm* blkJudge *as follows:*
On execution of $\mathsf{Judge}_{Msg}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \perp)$ *do the following:*

1. *If* $\mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) =$ `false` *then abort and output $\perp$ indicating an error.*
2. *Output $\perp$ if* $\exists 1 \leq i \leq |m| : \mathsf{blkJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \perp, i) = \perp$.
   Output `San` *if* $\exists 1 \leq i \leq |m| : \mathsf{blkJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \perp, i) = \mathtt{San}$.
   Output `Sig` *if* $\exists 1 \leq i \leq |m| : \mathsf{blkJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \perp, i) = \mathtt{Sig}$.

As also the blkJudge *algorithm correctly identifies the accountable party on an empty proof $\pi = \perp$, it is not a problem for the simulation that* Judge_{Msg} *was also only given $\perp$. Combined with the previous Theorems 6 and 7 the behaviour of the message-level properties follow.* □

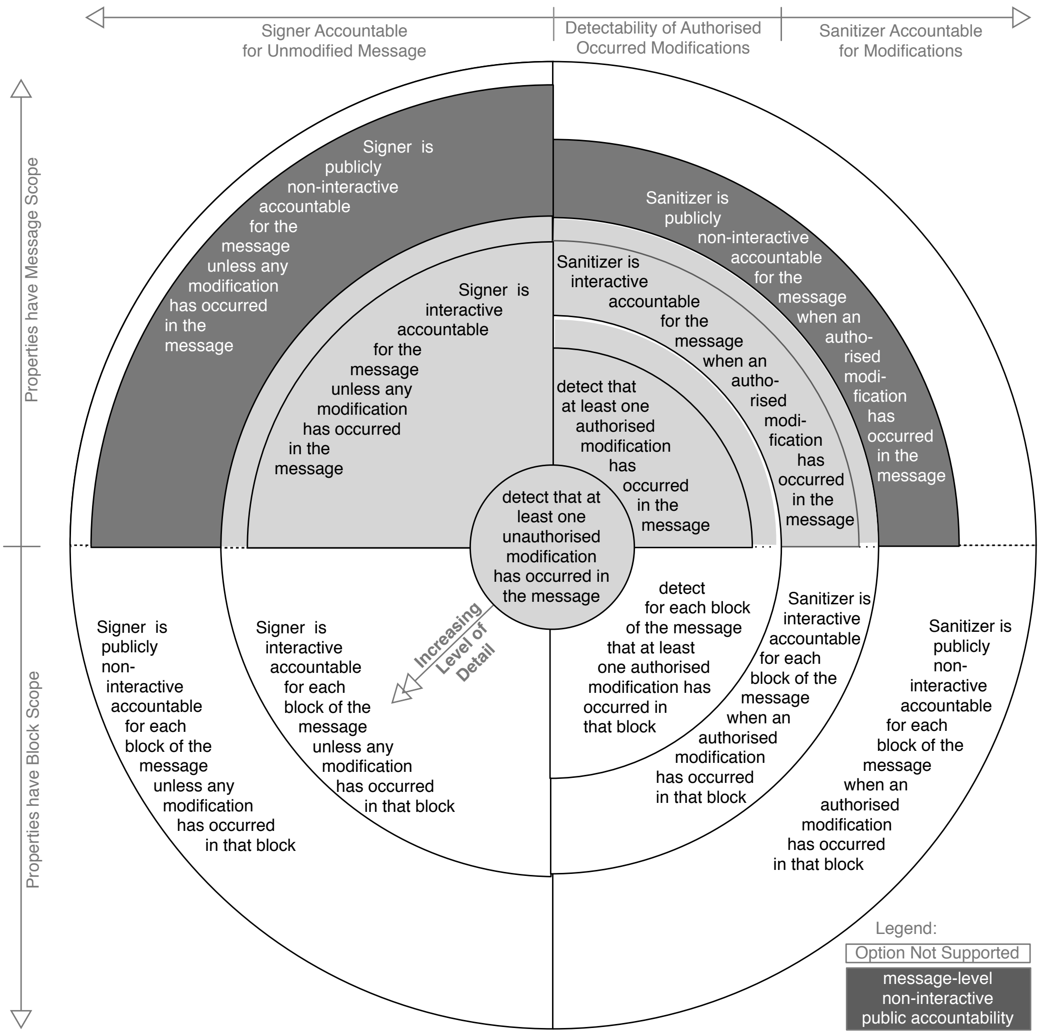

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Increasing Level of Detail
detect for each block of the message that at least one authorised modification has occurred in that block
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
message-level non-interactive public accountability

Figure 76. Visual representation of schemes that support forms of *non-interactive public* accountability of both the Signer and the Sanitizer on *message-level*

13.1.6. Visualisation of Non-interactive Public Accountability

The impact of the property message-level and block-level non-interactive public accountability can be shown using the visual representation of the integrity protection.[785]

13.1.6.1. Visualisation of message-level non-interactive public accountability

In Fig. 76 the impact of the property message-level non-interactive public accountability is shown. This property, as it is message-level is on the upper half. Due to its accountability being non-interactive public (PUB) it goes furthest to the edge of the circle.

[785] In general, the properties protection is depicted by marking the areas grey. From the centre circle towards the outside the visualisation shows an increasing protection scope that includes inner levels. Therefore the outermost level is marked in dark grey while the included levels are marked in light grey.

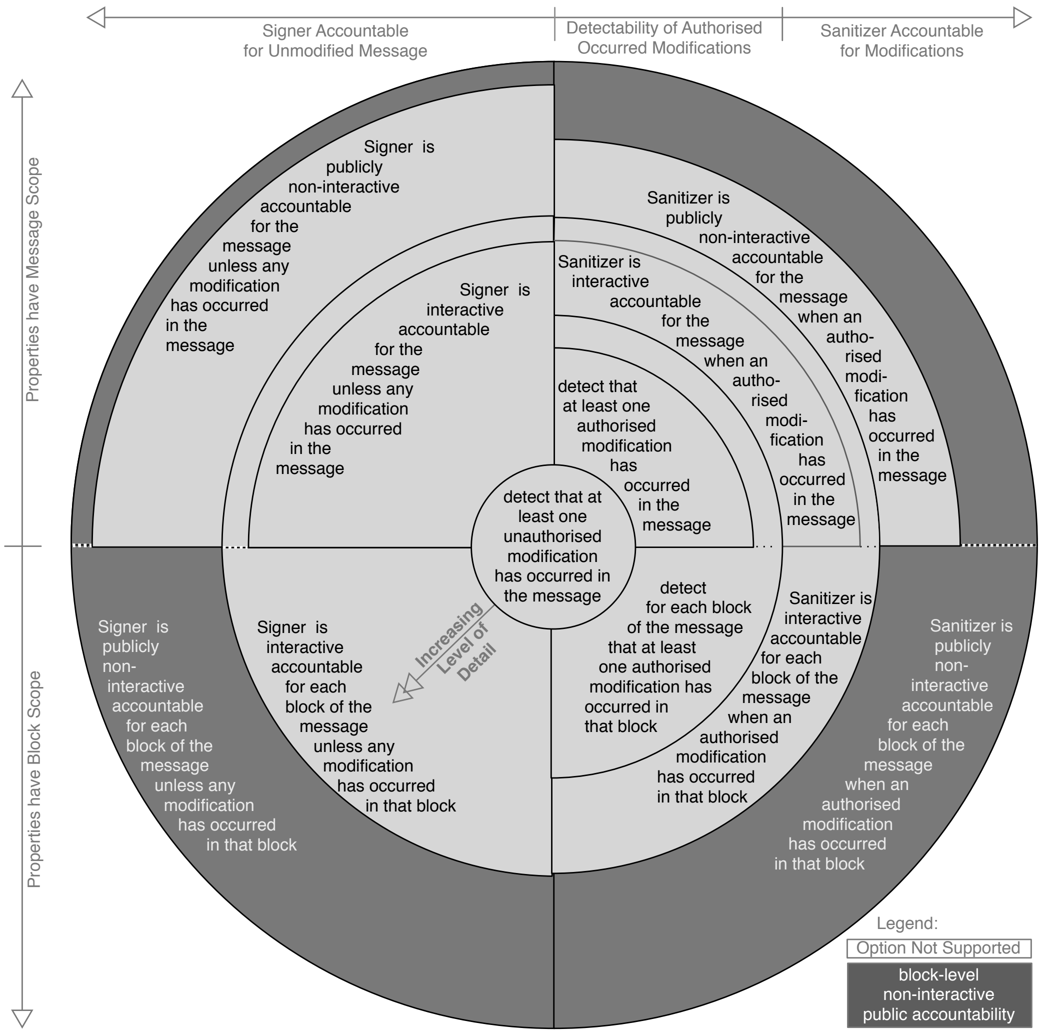

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an authorised modification has occurred in the message
Sanitizer is interactive accountable for the message when an authorised modification has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Increasing Level of Detail
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
block-level non-interactive public accountability

Figure 77. Visual representation of schemes that support forms of *public non-interactive* accountability of both the Signer and the Sanitizer on *block-level*, which *includes message-level* properties

13.1.6.2. Visualisation of block-level non-interactive public (PUB) accountability

In Fig. 77 the impact of block-level non-interactive public accountability is shown. This property is block-level, which is located on the lower half. As block-level properties can be used to construct message-level properties, this property includes layers of the upper half. With the accountability being non-interactive public (PUB) it goes furthest to the edge of the circle.

The visualisation, as intended to follow the focus on probative value, PUB accountability is depicted by reaching all the way to the outermost level/ring. When comparing the (block-level) non-interactive public accountability (PUB) from Fig. 77 with the interactive and non-public, but correctly-executed, accountability (INT) version as depicted in Fig. 81 on page 325, the contrast between the properties is also visually clearly differentiable.

13.2. Proposed SSS property: Accountability (transparent or non-interactive public) on the level of groups of blocks

This property of accountability (transparent or non-interactive public) on the level of groups of blocks[786] is an extension and generalisation of the block-level accountability. Block-level accountability, in contrast to message-level accountability is a property that is individually defined on a single block of a message. It was assumed to be defined for all blocks of a message. The extension discussed here is to define accountability over groups of blocks. This yields an accountability property for each level: transparent or interactive non-public (INT) versus non-interactive public (PUB) accountability. This thesis terms this *accountability on the level of block or groups of blocks*.

13.2.1. Shortcoming 4 addressed by block-level accountability

Previous state-of-the-art accountability of existing transparent SSS allowed only discovering interactively with the Signer that this attack has taken place. Thus, it required an additional and successful interaction with the Signer before using potentially sanitized data. As motivated in the discussion of Shortcoming 4[787], a first mitigation was the newly introduced non-interactive public accountability (see Sec. 13.1) as this removes the need for the Verifier to interact with the Signer or Sanitizer to gain confirmation of accountability. However, this can still be message based, and as sanitization in many application scenarios is allowed on multiple blocks of the message the application would benefit from knowing exactly which of these blocks are still unchanged. Only then can the probative value for each of these blocks be identified. Moreover, the knowledge allows the application to assign different levels of trust to information whether it was from a sanitized or an unmodified blocks. To counter Shortcoming 4[787] the accountability property on the block-level in introduced as a new SSS property in this thesis.

13.2.2. Definition of block-level and group-of-blocks-level accountability

In this section, this thesis presents a more fine-grained scope for the accountability property: level of block.[788] Henceforth, this thesis attributes security properties with an additional *message-level*, *block-level* or *group-level* where such a distinguishing is needed. Accountability allows the Signer to prove to outsiders, i.e., any Verifier, if a message was original or touched by the semi-trusted party of the Sanitizer. When having accountability a scheme could offer accountability either being transparent or non-interactively public (PUB), following the distinction drawn up in Sec. 6.4. For simplicity, this section recalls the definition of non-interactive public accountability (PUB) and interactive non-public accountability (INT) — known as transparency.

Definition 179 : Non-Interactive Public Accountability (PUB)

> *A sanitizable signature scheme satisfies* ***non-interactive public accountability*** *(PUB), if and only if for a valid message-signature pair (m, σ), a third party can correctly decide whether (m, σ) originates from the* Signer *or from the* Sanitizer *without requiring an interaction with the* Signer*, the* Sanitizer *or any other party with access to secret keys, i.e. just from using public knowledge $(m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san})$.*

Definition 157 : Cryptographic Transparency

> ***Transparency*** *prevents third parties which have only access to a given valid message-signature pair (m, σ) and public information (e.g., pk_{sig} or pk_{san}) to decide which party (*Signer *or* Sanitizer*) is accountable for a given valid message-signature pair (m, σ).*
>
> *Notes for Definition 157:*
>
> *Note 1 Information leakage through the message itself is out of scope of Definition 157.*

[786] The results given in this section have in part already been published at ARES 2013 as joint work with *K. Samelin*, *H. de Meer* and *J. Posegga* [132] (see Appendix A publication n° 16).

[787] Shortcoming 4: No accountability on a the fine-grained scope of blocks, only on the scope of the message (see Sec. 12.4).

[788] Note, in the original joint publication this was denoted as "blockwise" [132] (see Appendix A publication n° 16). In this thesis this term got harmonised as block-level, without changing its meaning.

Note 2 *Transparency is cryptographically stronger than the cryptographic privacy notion (see Definition 155), i.e., transparency implies privacy [64].*

Note 3 *Third party refers here to any party that is neither Signer or Sanitizer, i.e. an entity in the role of the Verifier. The third party has no access to non-public information; especially not to the secret keys (sk_{sig} or sk_{san}) or output generated involving them, i.e. no access to the proof π for the given valid message-signature pair (m, σ).*

In existing schemes, the two properties were defined for the whole message. Both message-level properties can be made more precise and brought to the scope of blocks. This is in line with the high-level terminology and understanding: Each message is composed out of blocks in an MSS (see Definition 41 in Sec. 3.2).

The block-level version of the accountability property in high-level language is then as follows:

Definition 187 : Block-level Accountability

*A sanitizable signature scheme offers **block-level accountability**, if for all valid pairs (m, σ) and their blocks $m[i]$, a third party can always correctly decide if the given block/signature pair $(m[i], \sigma)$ originates from the Signer or from the Sanitizer.*

The enhancement for accountability on the level of blocks requires to introduce one additional algorithm denoted as blkJudge, instead of just Judge. Most notably, compared to the input of a judge algorithm Judge, blkJudge additionally demands as an input an index i and then returns `San` or `Sig`. blkJudge indicates which party is accountable for the i^{th} group of block(s). The latter mentioned groups of block(s) are the reason why this is referred to as *group-level*. On a high-level, this property can be described as follows:

Definition 188 : Group-Level Accountability

***Group-Level Accountability** allows a predefined set of entities to decide for all valid message-signature pairs (m, σ) using an additional protocol (e.g., the algorithm Proof outputs a proof π which allows the algorithm grpJudge to decide) if the given pair of a **group-of-block** from a given message m and the signature over that given message σ originates from the Signer or from the Sanitizer.*

Notes for Definition 188:

Note 1 *Definition 188's group-level accountability holds even in the presence of malicious signers or sanitizers.*

Note 2 *Definition 188 allows the algorithm grpJudge to be non-interactive and public, e.g., would work on an empty proof, but does not demand it. Hence, Definition 188 is compatible with non-interactive public accountable (PUB) as well as transparent — interactive and non-public accountable (INT) — schemes.*

Group-level definitions imply both the block-level and message-level notions. In detail, group-level accountability includes the existing definitions as a border case: The standard accountability notion for the message-level can be defined on the basis of group-level accountability, such that a Sanitizer is accountable for a complete message-signature pair (m, σ), if there exists at least one group i, for which the Sanitizer has taken accountability. Vice versa, if there exists no group for which the Sanitizer has taken accountability, the Signer's accountability for the complete message-signature pair (m, σ) follows. This retains the expected behaviour from the original accountability notion which was defined on the message-level by *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk* [64] (see Sec. 11.6.6).

Definition 189 : Groups of Block(s) (grp)

grp contains a set of sets which expresses which admissible block(s) $m[i]$ are grouped together. $\mathsf{grp} \subseteq 2^{\mathbb{N}}$. To simplify the algorithmic description every non-admissible block belongs to the special group $\mathsf{grp}[0]$. Elements of grp are required to be pairwise disjunct, i.e., $\forall i, j \geq 0$ with $i \neq j : \mathsf{grp}_i \cap \mathsf{grp}_j = \emptyset$. Moreover, $|\bigcup_{s_i \in grp} s_i| = |ADM|$ must hold. In other words, every admissible block belongs to exactly one group.

The notation of $\mathsf{grp}[i]$ *is used to denote the uniquely reversible concatenation of all block that belong to* $\mathsf{grp}[i]$*. The set is ordered by order of appearance of the ordered blocks. The cardinality of* grp*, i.e., the number of groups, is denoted as* γ*.*

It is assumed that ADM *and* grp *can always be correctly reconstructed from* σ*, which counteracts the attacks described by Gong, Qian, and Zhou [216].*

EXAMPLE

The following example is given to clarify the notation: Let a message m contain six blocks, such that $m = m[1]||m[2]||m[3]||m[4]||m[5]||m[6]$. Hence, $\ell = 6$. Then let $\mathsf{grp} = \{\{1,5\},\{3,4,6\}\}$. This means, that $\mathsf{grp}[1] = (m[1], m[5])$ and $\mathsf{grp}[2] = (m[3], m[4], m[6])$. The cardinality of above grp is $\gamma = 2$. Finally, $\mathsf{grp}[0] = (m[2])$ denotes the blocks that are non-admissible.

As the algorithm Judge only decides the party which is accountable for the complete message-signature pair, and blkJudge only for a single block, this thesis requires an additional algorithm able to derive it for each group. The additional algorithm grpJudge is defined as follows:

Group-level Judge: Algorithm grpJudge takes a valid message-signature pair (m, σ), the corresponding public keys $\mathsf{pk}_{\mathrm{sig}}$ and $\mathsf{pk}_{\mathrm{san}}$, the Signer's proof π, and the group index i. grpJudge outputs the accountable party for the blocks contained in group i (or $\perp$ in case of an error; omitted for brevity).

$$d \leftarrow \mathsf{grpJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \pi, i), \quad d \in \{\mathtt{San}, \mathtt{Sig}\}\,.$$

The security parameter is denoted as λ. It is assumed that ADM is always recoverable from any signature $\sigma \neq \perp$ by the grpJudge holding a genuinely created proof π that corresponds to $\mathsf{pk}_{\mathrm{sig}}$.

13.2.3. Group-level non-interactive public (PUB) accountability for SSS

Following in line of the previously newly introduced property of non-interactive public accountability (see Sec. 13.1), the group-level accountability property can also offer the public form of accountability (as distinguished from INT in Sec. 6.4). This requires the Verifier to need no interaction with the Signer, i.e., the following definition defines that the algorithm generally known as grpJudge decides correctly upon input of an empty proof, i.e., $\pi = \perp$.

For notational reasons the additional algorithm grpDetect is introduced. It is a non-interactive and public version of grpJudge and thus it does not even require the proof π as a parameter:

$$\mathsf{grpDetect}(1^\lambda, m, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, i) = \mathsf{grpJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \perp, i)\,.$$

In detail grpDetect is defined as follows:

Group-level Judge (PUB): Algorithm grpDetect takes as input the security parameter λ, a valid message-signature pair (m, σ), the corresponding public keys $\mathsf{pk}_{\mathrm{sig}}$ and $\mathsf{pk}_{\mathrm{san}}$, and the group index i. grpDetect outputs the accountable party for the blocks contained in group i (or $\perp$ in case of an error; omitted for brevity).

$$d \leftarrow \mathsf{grpDetect}(1^\lambda, m, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, i), \quad d \in \{\mathtt{San}, \mathtt{Sig}\}\,.$$

The security parameter is denoted as λ. It is assumed that ADM is always recoverable from any signature $\sigma \neq \perp$ by the grpDetect.

Definition 190 : Group-level Non-Interactive Public Accountability (PUB) for SSS

A SSS *together with an algorithm* blkJudge *is* ***group-level non-interactive public (PUB) accountable****, if for any efficient algorithm* $\mathcal{A}$ *the probability that the experiment* $\text{Group-Pub-Acc}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ *given in Fig. 78 returns* 1 *is negligible (as a function of* λ*).*

```
**Experiment** $\mathsf{Group\text{-}Pub\text{-}Acc}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\text{sig}}, \mathsf{sk}_{\text{sig}}) \leftarrow \mathsf{KGen}_{\text{sig}}(1^{\lambda})$
  $(\mathsf{pk}_{\text{san}}, \mathsf{sk}_{\text{san}}) \leftarrow \mathsf{KGen}_{\text{san}}(1^{\lambda})$
  $\mathcal{Q}_{\text{sig}} \leftarrow \emptyset$
  $\mathcal{Q}_{\text{san}} \leftarrow \emptyset$
  $(\mathsf{pk}^*, m^*, \sigma^*, q^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^{\lambda},\cdot,\mathsf{sk}_{\text{sig}},\cdots),\mathsf{Sanit}(1^{\lambda},\cdots,\mathsf{sk}_{\text{san}})}(1^{\lambda}, \mathsf{pk}_{\text{san}}, \mathsf{pk}_{\text{sig}})$
    Let $(m_i, \mathsf{ADM}_i, \mathsf{pk}_{\text{san},i}, \mathsf{grp}_i)$ and $(m_i, \sigma_i)$ for $i = 1, 2, \ldots, m$
      be the queries and answers to and from oracle Sign.
    For every query to Sign, do
      $\mathcal{Q}_{\text{sig}} \leftarrow \mathcal{Q}_{\text{sig}} \cup \{(\mathsf{pk}_{\text{san,i}}, \mathsf{ADM}_i, \mathsf{grp}_i, \mathsf{grp}_i[0], (m_i[k], k)\}$
      for all $k \in \{1, 2, \ldots, \ell_i\}$.
    Let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk}_{\text{sig,j}})$ and $(m'_j, \sigma'_j)$ for $j = 1, 2, \ldots, m'$
      be the queries and answers to and from oracle Sanit.
    For every query to Sanit, do
      $\mathcal{Q}_{\text{san}} \leftarrow \mathcal{Q}_{\text{san}} \cup \{(\mathsf{pk}_{\text{sig,j}}, \mathsf{ADM}_j, \mathsf{grp}_j, \mathsf{grp}_j[0], (m_j[l], j)\}$
      for all $l \in \{1, 2, \ldots, \ell_j\}$.
    return 1, if
      $\mathsf{Verify}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}_{\text{sig}}, \mathsf{pk}^*) = \texttt{true}$, and
      $\mathsf{grpDetect}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}_{\text{sig}}, \mathsf{pk}^*, q^*) = \texttt{Sig}$, and
      $(\mathsf{pk}^*, \mathsf{ADM}_i^*, \mathsf{grp}^*, \mathsf{grp}^*[0], (m^*[q^*], q^*)) \notin \mathcal{Q}_{\text{sig}}$
    return 1, if
      $\mathsf{Verify}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\text{san}}) = \texttt{true}$, and
      $\mathsf{grpDetect}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\text{san}}, q^*) = \texttt{San}$, and
      $(\mathsf{pk}^*, \mathsf{ADM}^*, \mathsf{grp}^*, \mathsf{grp}^*[0], (m^*[q^*], q^*)) \notin \mathcal{Q}_{\text{san}}$
  return 0
```

Figure 78. Group-level Non-Interactive Public (PUB) Accountability Experiment for SSS

13.2.4. Group-level interactive non-public (INT / transparent) accountability for SSS

Accountability, still with a scope of groups of block(s), can fully preserve transparency. Note, extra care was taken when describing the general notion of block-level accountability to keep it general (see Definition 187) and to make it overarch both forms of accountability.

Definition 191 : Group-level Interactive Non-Public Signer-Accountability (INT) for SSS

*An SSS is **group-level interactive non-public Signer-accountable**, if for any efficient algorithm $\mathcal{A}$ the probability that the experiment Group-Signer-Acc$_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 79 returns 1 is negligible (as a function of λ). Basically, to win the game the adversary has to generate a tuple $(pk^*, m^*, \sigma^*, \pi^*)$, which leads grpJudge to decide that the Sanitizer is accountable for a group grp$[q] \in m^*$, while it is not.*

Definition 192 : Group-level Interactive Non-Public Sanitizer-Accountability for SSS

*An SSS is **group-level interactive non-public Sanitizer-accountable**, if for any efficient algorithm $\mathcal{A}$ the probability that the experiment Group-Sanitizer-Acc$_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 80 returns 1 is negligible (as a function of λ). Basically, to win the game the adversary has to generate a tuple (pk^*, m^*, σ^*) for which* Proof *generates a proof π which leads grpJudge to decide that the Signer is accountable for a group grp$[q] \in m^*$, while it is not.*

Definition 193 : Group-level Interactive Non-Public (INT) Accountability for SSS

*A SSS is **group-level interactive non-public (INT) accountable**, if it is group-level interactive non-public Signer-accountable and group-level interactive non-public Sanitizer-accountable.*

```
Experiment Group-Signer-Acc$^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^{\lambda})$
  $b \leftarrow \{0,1\}$
  $(\mathsf{pk}^*, \pi^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sanit}(1^{\lambda},\cdots,\mathsf{sk}_{\mathrm{san}})}(1^{\lambda}, \mathsf{pk}_{\mathrm{san}})$
    Let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk}_{\mathrm{sig,j}})$ and $(m'_j, \sigma'_j)$ for $j = 1,2,\ldots,k$
    be the queries and answers to and from the oracle Sanit.
    For each query to Sanit, do
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mathsf{pk}_{\mathrm{sig,j}}, \mathsf{ADM}_j, \mathsf{grp}_j, \mathsf{grp}_j[0], (m_j[l], l))\}$
      for all $l \in \{1,2,\ldots,\ell_j\}$.
  return 1, if:
    $\mathsf{Verify}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}) = \mathtt{true}$, and
    $\mathsf{grpJudge}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}, \pi^*, q^*) = \mathtt{San}$, and
    $(\mathsf{pk}^*, \mathsf{ADM}^*, \mathsf{grp}^*, \mathsf{grp}^*[0], (m^*[q^*], q^*) \notin \mathcal{Q}$ // was never queried to Sanit
```

Figure 79. Group-level Interactive Non-Public Signer-Accountability (INT) Experiment for SSS

```
Experiment Group − Sanitizer − Acc$^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^{\lambda})$
  $b \leftarrow \{0,1\}$
  $\mathcal{Q} \leftarrow \emptyset$
  $(\mathsf{pk}^*, m^*, \sigma^*, q^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^{\lambda},\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Proof}(1^{\lambda},\mathsf{sk}_{\mathrm{sig}},\cdots)}(1^{\lambda}, \mathsf{pk}_{\mathrm{sig}})$
    Let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk}_{\mathrm{san},j}, \mathsf{grp}_j)$ and $(m_j, \sigma_j)$
      be the queries and answers to and from oracle Sign.
    For each query to Sign, do
      $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mathsf{pk}_{\mathrm{san},j}, \mathsf{ADM}_j, \mathsf{grp}_j, \mathsf{grp}_j[0], (m_j[l], l))\}$
      for all $l \in \{1,2,\ldots,\ell_j\}$.
  $\pi \leftarrow \mathsf{Proof}(1^{\lambda}, \mathsf{sk}_{\mathrm{sig}}, m^*, \sigma^*, \{(m_i, \sigma_i) | 0 < i \leq k\}, \mathsf{pk}^*)$
  return 1, if:
    $\mathsf{Verify}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*) = \mathtt{true}$, and
    $\mathsf{grpJudge}(1^{\lambda}, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*, \pi, q^*) = \mathtt{San}$, and
    $(\mathsf{pk}^*, \mathsf{ADM}^*, \mathsf{grp}^*, \mathsf{grp}^*[0], (m^*[q^*], q^*) \notin \mathcal{Q}$
```

Figure 80. Group-level Interactive Non-Public Sanitizer-Accountability (INT) Experiment for SSS

13.2.5. Relation to existing properties

The existing definitions for message- and block-level accountability are contained as border-cases in the new group-level ones.

Theorem 9 : Group-level Signer-Accountability ⇒ Block-level Signer-Accountability ⇒ Message-level

Signer-Accountability] Every SSS which is group-level Signer-accountable, can be used to construct a scheme which is (block-|message-level) Signer-accountable.

Proof 9

To emulate block-level Signer-accountability for a message with ℓ blocks the SSS is used with ℓ groups, each group contains as a single element the corresponding block. If the group-level Signer-accountable SSS indicates the accountability for the group i this emulates block-level Signer-accountability for blocks with the identifier i.

To emulate message-level Signer-accountability for a message with ℓ blocks the SSS is used with 1 group which contains all ℓ block. If the group-level Signer-accountable SSS indicates the accountability for the group i this emulates block-level Signer-accountability for blocks with the identifier i. □

Theorem 10 : Group-level Sanitizer-Accountability ⇒ Sanitizer-Accountability

Every SSS which is group-level Sanitizer-accountable, is also Sanitizer accountable.

Proof 10

> *Sanitizer-accountability can be emulated analogous to Signer-accountability. The proof is thus analogue to the one shown in Proof 9.* □

13.2.6. Visual of group-level accountability

The visual representation of the integrity protection, as introduced in Sec. 6.9 does not contain an extra visual for group-level properties. A possible visualisation would add an additional new outermost ring; because group-level properties can emulate block-level and message-level properties and would thus assimilate those inner rings of the property-slices.

13.2.6.1. Visual of group-level non-interactive public (PUB) accountability

As mentioned previously in Sec. 13.2.5 a group-level property includes the same property on block-level. As block-level properties — in the visualisation on the bottom half — encircle the message-level properties — shown on the upper half — the visual representation of a scheme offering group-level non-interactive public (PUB) accountability would look similar to the one for the block-level property with an additional ring to the outside for the group-level. The block-level non-interactive public (PUB) accountability was shown in Fig. 77 on page 318. Thus, a new figure is not provided; Fig. 77 was already shown when the block-level non-interactive public (PUB) accountability was introduced Sec. 13.1.6.2.

13.2.6.2. Visual of block- and group-level interactive non-public (INT / transparent) accountability

The visualisation shows that block-level properties include message-level ones (see Sec. 13.2.5). However, the visualisation introduced in Sec. 6.9 does not contain extra rings to visualise group-level accountability properties. An interactive non-public (INT) group-level accountability would include both: block-level and message-level accountability, but both only at INT level. The visualisation was made to differentiate between transparent — thus interactive non-public (INT) accountable — and non-interactive public accountable (PUB) schemes. For this, the outermost ring in the visualisation indicates that PUB includes INT. A scheme that offers group-level interactive non-public (INT) accountability would thus look similar to the block-level property shown in Fig. 81 on page 325. The outermost ring is not reached by group-level INT accountability because this already visualises the level of PUB accountability. A possible visualisation would require to extend the visual with an additional ring inside of, i.e., below, PUB. This ring would need to split INT accountability — depicted in the two lower half quadrants — into group-level and block-level. However, to keep the simplicity of the visualisation, no such extension was introduced and no figure is provided for group-level interactive non-public (INT) accountability. Instead, Fig. 81 on page 325 shows the block-level interactive non-public (INT) accountability as it was not previously depicted.

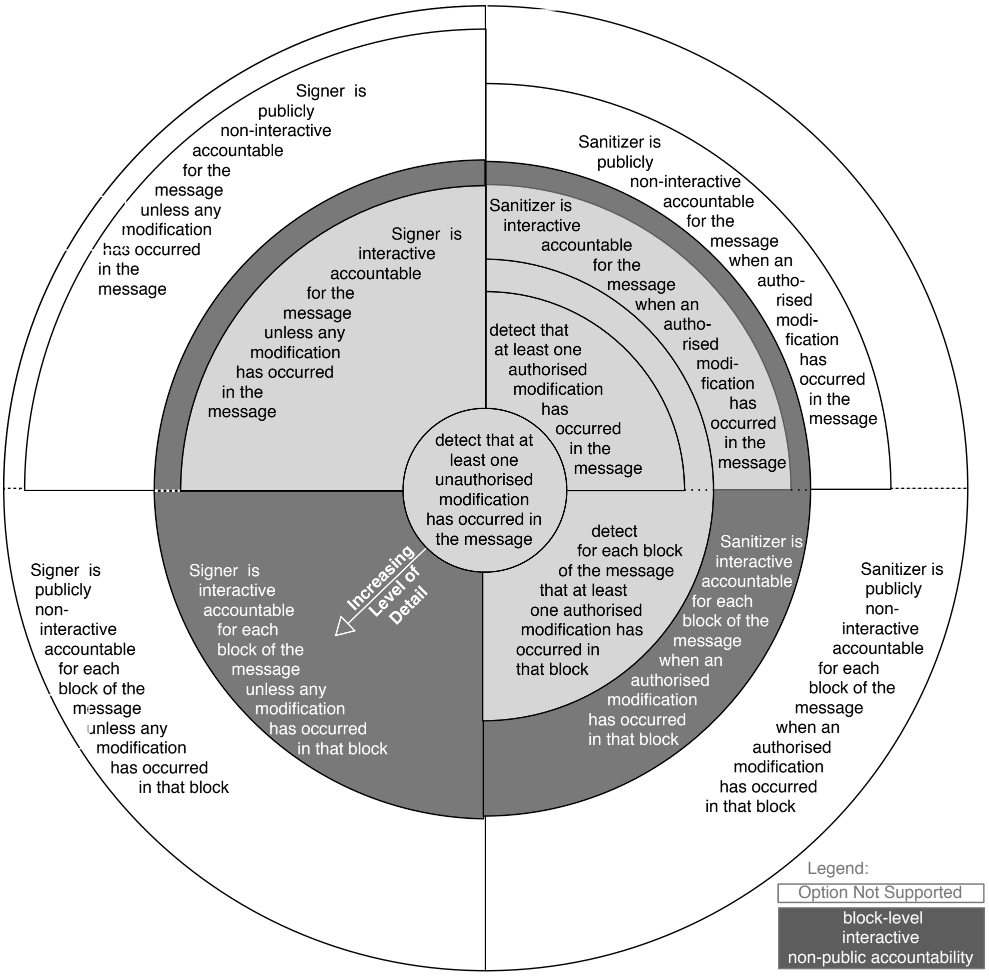

Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Increasing Level of Detail
Legend:
Option Not Supported
block-level interactive non-public accountability

Figure 81. Visual representation of the schemes that support block-level interactive non-public (INT / transparent) accountability

13.3. Proposed SSS property: Strengthened unlinkability

In the following the notion of unlinkability is strengthened[789]. In the context of sanitizable signatures, the notion of unlinkability captures that two sanitized messages cannot be linked to having the same original message-signature pair. For group signatures [121], in turn, the unlinkability definition corresponds to the anonymity of the Signer. Anonymity of the Signer in the context of sanitizable signatures can be seen as related to hide from the Verifier whether the Signer or the Sanitizer is accountable, this property already exists and is called transparency. In the area of malleable signatures the view of cryptographic properties is message-centred, while the way of thinking in the area of group signatures is more Signer-centred — after all, the word “group” refers to a group of signers, not to a group of messages. To avoid confusion due to the historical evolution of the properties’ names in the two areas, this thesis sticks with the nomenclature from the MSS literature as introduced by *Brzuska et al.* [67].

The property of strengthened unlinkability[789] is realised by the $unlinkable\mathcal{SSS}$ scheme that is presented in Definition 200. As privacy properties are not part of the visualisation introduced in Sec. 6.9, this increase in the privacy protection through a strengthening of unlinkability does not show in the visualisation.Thus, no visualisation is provided.

[789] The strengthening of the unlinkability property and the results given in this section have in part already been published at EuroPKI 2013 as joint work with *C. Brzuska* and *K. Samelin* [69] (see Appendix A publication nº 17).

13.3.1. Shortcoming 1 and Shortcoming 3 addressed by strengthened unlinkability

Unlinkability is a stronger privacy notion as it implies standard privacy following *Brzuska, Fischlin, Lehmann, and Schroeder* in [67][790]. Recall, an unlinkable scheme makes it hard to link two sanitized messages, i.e., it is hard to detect that they were derived from the same message-signature pair. In the hospital patient record database example, it might as well happen that the involved entities derive two different sanitized documents from the original patient's record. This happens when different parts from the same signed patient record are removed, e.g., to create a version for the accountant and another one for the reception desk personnel. A secure solution must prevent inferring information about the original record by combining the two sanitized documents, as this would violate the patients' privacy concerns as well as data-protection regulations such as HIPAA [484]. Thus, in such application domains [67], it is important for sanitizable signature schemes to achieve unlinkability.

The existing unlinkability definition by *Brzuska et al.* was presented at PKC '10 [67]. This thesis refers to this notion henceforth as standard unlinkability. Stronger notions of *statistical* unlinkability have been discussed for RSS by *Ahn* et al. at TCC '12 [5], or — again only for RSS — stronger notions of privacy including unlinkability as in [15, 16, 136]. The strengthening is done by allowing the adversary to know the Signer's secret key. Hereby, unlinkability of two sanitized messaged shall still hold even if the Signer is malicious or maybe just buggy. Intuitively, this captures that unlinkability holds as long as the Sanitizer is honest, even if the Signer happens to be dishonest. One might argue that a malicious Signer can always break any privacy or unlinkability property, as an adversarial Signer knows which messages it signed and thus, it can always recover the originally signed message. However, there exist intermediary stages, for example, if the Signer is buggy, uses weak randomness, or loses its secret key. In these cases, the new definition turns out to be robust, i.e., even if the Signer loses its secret key, unlinkability and therefore privacy is preserved after a sanitization.

In a sense, strengthened unlinkability covers the equivalent of forward secrecy for the case of key exchange: there, previous sessions remain secure when long-term secrets are lost [114]. Actually, strengthened unlinkability even achieves a stronger notion of secrecy than key exchange can attain: Unlinkability is even preserved when the Signer loses his secret key *before* signing the message and even when using some form of "bad" secret key specified by the adversary.

The strengthened unlinkability addresses Shortcoming 1 and Shortcoming 3.

13.3.2. Definition of strengthened unlinkability

Definition 194 : Strengthened Unlinkability

> *A sanitizable signature scheme* SSS *is* ***unlinkable****, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment* $\mathsf{SUnlinkability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ *given in Fig. 82 returns* 1 *is negligibly close to* $\frac{1}{2}$ *(as a function of* λ*).*

In the experiment from Fig. 82 the adversary has to guess which of the two message-signature pairs that were provided as inputs to LoRSanit was chosen to be sanitized.

Please note, *Brzuska et al.* [64] require that $\mathsf{MOD}_i \subseteq \mathsf{ADM}_i$ is true. Both LoRSanit oracles[791] check this condition for all queries as otherwise a trivial attack vector exists. In particular, if $\mathsf{MOD}_i \not\subseteq \mathsf{ADM}_i$ yields that Sanit outputs $\perp$ then this is easily distinguishable from a "fresh" signature.

A comparison of the new definition of unlinkability in Fig. 82 with the original definition of [67] depicted for easy comparison in Fig. 83 yields the following: The LoRSanit oracle got altered such that it does not specify the Signer by having a fixed Signer key $\mathsf{pk}_{\mathrm{sig}}$. In turn, now, the adversary chooses $\mathsf{pk}_{\mathrm{sig}}$. In Definition 194, there are no signing and proof oracles — the reason is that the adversary can now simulate those by itself.

[790] For details on the existing definition see Sec. 11.6.5.

[791] LoRSanit oracle from Fig. 83 and from Fig. 82. Note that on the game from Fig. 83 was used to derive the harmonised definition given in Sec. 11.6.5 on page 269 with Fig. 54.

```
Experiment $\mathsf{SUnlinkability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
  $b \leftarrow \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}}),\mathsf{LoRSanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{san}})$
    where oracle LoRSanit on input of:
    $m_{0,i}, \mathsf{MOD}_{0,i}, \sigma_{0,i}, m_{1,i}, \mathsf{MOD}_{1,i}, \sigma_{1,i}, \mathsf{pk}_{\mathrm{sig},i}$
    //ADM needs to be recoverable from all $\sigma$
    if $\mathsf{ADM}_{0,i} \neq \mathsf{ADM}_{1,i}$, return $\perp$
    if $\mathsf{MOD}_{0,i} \not\subseteq \mathsf{ADM}_{0,i}$, or $\mathsf{MOD}_{1,i} \not\subseteq \mathsf{ADM}_{1,i}$, or $\mathsf{MOD}_{0,i}(m_{0,i}) \neq \mathsf{MOD}_{1,i}(m_{1,i})$, return $\perp$
    if $\mathsf{Verify}(1^\lambda, m_{0,i}, \sigma_{0,i}, \mathsf{pk}_{\mathrm{sig},i}, \mathsf{pk}_{\mathrm{san}}) \neq \mathtt{true}$ or $\mathsf{Verify}(1^\lambda, m_{1,i}, \sigma_{1,i}, \mathsf{pk}_{\mathrm{sig},i}, \mathsf{pk}_{\mathrm{san}}) \neq \mathtt{true}$, return $\perp$
    return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_{b,i}, \mathsf{MOD}_{b,i}, \sigma_{b,i}, \mathsf{pk}_{\mathrm{sig},i}, \mathsf{sk}_{\mathrm{san}})$
  return 1, if $a = b$
```

Figure 82. Strengthened Unlinkability Experiment for SSS

```
Experiment $\mathsf{Unlinkability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{pk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
  $b \leftarrow \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}})}_{\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots)\mathsf{LoRSanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}},\mathsf{sk}_{\mathrm{sig}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
    where oracle LoRSanit on input of
    $m_{0,i}, \mathsf{MOD}_{0,i}, \sigma_{0,i}, m_{1,i}, \mathsf{MOD}_{1,i}, \sigma_{1,i}$:
      //ADM needs to be recoverable from all $\sigma$
      if $\mathsf{ADM}_{0,i} \neq \mathsf{ADM}_{1,i}$, return $\perp$
      if $\mathsf{MOD}_{0,i} \not\subseteq \mathsf{ADM}_{0,i}$, return $\perp$
      if $\mathsf{MOD}_{1,i} \not\subseteq \mathsf{ADM}_{1,i}$, return $\perp$
      if $\mathsf{MOD}_{0,i}(m_{0,i}) \neq \mathsf{MOD}_{1,i}(m_{1,i})$, return $\perp$
      if $\mathsf{Verify}(1^\lambda, m_{0,i}, \sigma_{0,i}, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}) \neq \mathtt{true}$, return $\perp$
      if $\mathsf{Verify}(1^\lambda, m_{1,i}, \sigma_{1,i}, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}) \neq \mathtt{true}$, return $\perp$
      return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_{b,i}, \mathsf{MOD}_{b,i}, \sigma_{b,i}, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$
  return 1, if $a = b$
```

Figure 83. Standard Unlinkability Experiment for SSS by *Brzuska et al.* [67]

13.3.3. Relation to existing properties

In the following this thesis states the proof that the new notion of strengthened unlinkability is strictly stronger than the original one by *Brzuska et al.* [67].

Theorem 11 : Strong Unlinkability $\Rightarrow$ Unlinkability (in the context of secure SSS)

Any strongly unlinkable SSS is also unlinkable. The converse is not true.

Proof 11

Let SSS be a strongly unlinkable sanitizable signature scheme. The following proves via reduction that SSS is also unlinkable. Let $\mathcal{A}$ be an adversary against the unlinkability of SSS. Using $\mathcal{A}$ allows to construct an adversary $\mathcal{B}$ against the strong unlinkability of SSS as follows. In the beginning of the game, $\mathcal{B}$ relays the sanitizer's public key to $\mathcal{A}$ and runs the key generation algorithm of the signer and returns the signer's public key pk_{sig} to $\mathcal{A}$. The signer's secret key is used to answer all queries that $\mathcal{A}$ makes to Sign and Proof. The remaining queries are queries to the Sanit oracle, which $\mathcal{B}$ simply relays, and queries to the LoRSanit oracle which $\mathcal{B}$ prepends with the signer's public key pk_{sig} and queries to its own LoRSanit oracle. In the end, $\mathcal{A}$ returns a bit that $\mathcal{B}$ returns as well. As the simulation is perfect, $\mathcal{B}$'s advantage against strong unlinkability is as big as $\mathcal{A}$'s advantage against unlinkability.

For separation of the two unlinkability notions a (contrived) scheme can be constructed that fulfils the security requirements of the original security model by Brzuska et al. [67] but is insecure in the new model.

Let $\mathsf{SSS} = (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$ *be a secure sanitizable signature scheme. To obtain a counterexample, the scheme is adjusted as follows:*

- KGen'_{sig} *works as* KGen_{sig}*, except that it appends a* 1 *to the public key returned by* KGen_{sig}
- KGen'_{san} *works as its original counterpart*
- Judge'*,* Sign'*, and* Verify' *work as their original counterparts, but cut of the last bit of* pk_{sig}
- Sanit' *is the same as* Sanit *with one exception: If* pk_{sig} *ends with a* 0*, it proceeds as normal, while cutting of the last bit of* pk_{sig}*. Otherwise, it outputs the original signature and message, instead of sanitizing it.*

The here stronger adversary can choose pk_{sig} *by running* KGen'_{sig}*. The adversary now wins by replacing the trailing* 1 *with a* 0*. However, the scheme is fully secure in the original definition where an adversary was not able to influence* pk_{sig}*. Here, it does not matter whether or not the keys are indistinguishable. All other properties are not affected and are inherited from the original* SSS*. Note, this scheme is also still correct as the message-signature pair that was left untouched by the modified* Sanit' *still verifies, while also* $\mathsf{MOD}_1(m_1) = \mathsf{MOD}_2(m_2)$ *yields.* □

As this stronger notion of unlinkability implies the original notion of unlinkability, all known implications still hold. In particular, this strengthened unlinkability thus implies standard privacy [67][792]. Thus, the following relations hold:

$$\text{strengthened unlinkability} \;\Rightarrow\; \text{(standard) unlinkability} \Rightarrow\; \text{standard privacy}\,.$$

Note, unlinkability implies privacy, while non-interactive public (PUB) accountability implies accountability (see Theorem 4 given in Sec. 13.1.2 on page 310). Recall the following separation by *Brzuska et al.* [67], which also applies for the strengthened unlinkability definition, as the latter implies the original definition of unlinkability in [67], as shown above.

Theorem 12 : Unlinkability ⇏ Transparency

There exists a scheme which is unlinkable, but not transparent.

The proof is re-stated from the work of *Brzuska et al.* [66] for readability:

Proof 12

Theorem 12 is proven by modifying an arbitrary existing unlinkable scheme SSS*. Let* $\mathsf{SSS} = (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$ *be an unlinkable sanitizable signature scheme. The scheme is altered into* SSS' *as follows:*

- $\mathsf{KGen}'_{sig} = \mathsf{KGen}_{sig}$*,* $\mathsf{KGen}'_{san} = \mathsf{KGen}_{san}$*, i.e., the key generation algorithms remain unchanged*
- Sign' *is the same as* Sign *with one exception; it appends a* 0 *to the final signature, i.e.,* $\sigma' = \sigma||0$
- Sanit' *is the same as* Sanit *with one exception; it removes the last bit from the signature, while it appends a* 1 *to the resulting signature*
- Verify' *is the same as* Verify *with one exception; it removes the last bit from the signature before proceeding with verification*

[792] *Brzuska et al.* showed "[...] that unlinkability does not follow from any of the other security requirements. Then we prove that unlinkability implies privacy, and finally discuss that unlinkability does not imply any of the other properties." [67]

- *Proof' and Judge' work essentially the same as their original counterparts while cutting of the last bit from the signature before proceeding*

Clearly, an outsider can now distinguish between signatures generated by the Signer *(last bit* 0*) and the* Sanitizer *(last bit* 1*), while all other properties, in particular unlinkability, are preserved.* □ *[66]*

13.4. *pubacc$\mathcal{SSS}$* concept: (Weak | standard) (block-level | message-level) non-interactive public accountability

In the following the scheme *pubacc$\mathcal{SSS}$* with non-interactive public accountability for the message and blocks is presented[793].

13.4.1. *pubacc$\mathcal{SSS}$* extended security properties

The secure sanitizable signature scheme *pubacc$\mathcal{SSS}$* achieves the following security properties (**bold** indicates a new security property introduced in this thesis):

- Unforgeability according to Definition 152, and
- Immutability according to Definition 154, and
- Standard Privacy following *Brzuska et al.* according to Definition 156, and

either **Non-interactive public accountability for the message** according to Definition 180,

or **Non-interactive public accountability for each individual block** according to Definition 190.

By definition the public form of accountability requires that third parties can immediately decide whether a message has been issued by the Signer or the Sanitizer. *pubacc$\mathcal{SSS}$* is for situations were transparency is not desirable. Especially, the need to uphold transparency cryptographically becomes obsolete if a sanitization is already detectable from obvious changes to the message. *pubacc$\mathcal{SSS}$* allows to achieve an increased legal probative value, as the evaluation in Chapter 17 concludes. Hence, *pubacc$\mathcal{SSS}$*'s main goal is to achieve the — thus preferable — non-interactive public form of accountability, while preserving unforgeability, immutability and privacy. The latter is still achieved in the strength as defined in Definition 156 following *Brzuska et al.* [64]. Moreover, *pubacc$\mathcal{SSS}$* achieves even the stronger notion of the property as introduced by *Krenn et al.* [294][794]

13.4.2. Adjustment of privacy definition for block-level and group-level *pubacc$\mathcal{SSS}$*

A cryptographic challenge is to uphold privacy even in the light of the adversary's ability to detect that a sanitizations has taken place due to non-interactive public accountability.

The empty modification instruction must make the Sanitizer accountable to uphold privacy while allowing non-interactive public accountability. It is required that MOD $= \emptyset$ does change the accountability. It will be needed to be interpreted as if $(i, m[i]') \in$ MOD with $m[i]' \leftarrow m[i]$ for blocks with an empty MOD or as if $(i, m') \in$ MOD with $m' \leftarrow m$ for messages. This got introduced as 'touching a message' in Definition 47 on page 49.

[793] The scheme *pubacc$\mathcal{SSS}$* with non-interactive public accountability for the message and blocks have in part already been published at EuroPKI 2012 as joint work with *C. Brzuska* and *K. Samelin* [68] (see Appendix A publication n° 11). Early joint results on block-level properties were also published in a technical report with *K. Samelin*, *J. Posegga* and *H.d. Meer* [428] (see Appendix A publication n° 27).

[794] The authors state the following: "The construction given in [68] achieves non-interactive public accountability. Due to the determinism and uniqueness of the signature schemes deployed, their scheme already fulfills our strong definitions without any modifications."[795] [796] [294]

As already discussed in Sec. 13.1.4 this difference is of paramount importance for non-interactive public accountability not to negatively impact on privacy. Due to the identity function being flagged as a sanitization by blkJudge, blkJudge, a Verifier and an adversary will not learn additional information about the original message from knowing that it was modified.

EXAMPLE

Consider this example showing how the adversary can win the game of privacy: On input of $m_0(H||e||l||l||o)$ with an empty modification instruction and $m_1 = (H||a||l||l||o)$ with the instruction to change the 2^{nd} element from 'a' into 'e', the adversary can use the public $\mathsf{Judge}(1^\lambda, m', \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \perp)$ to identify which input message was used to generate the signed $m' = (H||e||l||l||o)$ presented by the oracle. If a block-level non-interactive public accountability is offered the oracle must ensure that the adversary submits also a modification instruction directed to the 2^{nd} element, e.g. $m_0 = (H||e||l||l||o)$ with the instruction to modify the 2^{nd} element to 'e'.

This is already covered for message-level non-interactive public accountability in the privacy game from the harmonised standard privacy definition given in Sec. 11.6.3 based on *Brzuska et al.* [64] including additions from *Gong et al.* [216] and *Krenn et al.* [294]. As long as the algorithms Sanit and Judge on an empty proof work such that Judge correctly attributes `San` after an empty modification instruction, privacy can be retained even with message-level non-interactive public accountability. When block-level non-interactive public accountability is offered the oracle must be adjusted to exclude cases where the adversary would submit empty modifications for a different block: For example the adversary can win the game of privacy on input of $m_0(H||e||l||l||o)$ with an empty modification instruction for the third block and $m_1 = (H||a||l||l||o)$ with the instruction to change the 2^{nd} block from 'a' into 'e', because the adversary can use the public $\mathsf{blkJudge}(1^\lambda, m', \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \perp, 2)$ to identify which input message was used to generate the signed $m' = (H||e||l||l||o)$ presented by the oracle. The adjustments needed are given in Fig. 84. For better readability the additions compared to Fig. 51 from the harmonised standard privacy are underlined in Fig. 84 while retaining all original oracle functionality[797].

```
**Experiment** $\mathsf{Privacy\text{-}blkJudge}^{\mathsf{SSS}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{pk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Sanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}})}_{\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{LoRSanit}(1^\lambda,\ldots,\mathsf{sk}_{\mathrm{sig}},\mathsf{sk}_{\mathrm{san}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
    where oracle LoRSanit on input of $m_0, \mathsf{MOD}_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$ // resulting $m'$ is the same
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{ADM} \not\subseteq m_0$ or $\mathsf{ADM} \not\subseteq m_1$ or $\mathsf{MOD}_0(\mathsf{ADM}) \not\subseteq \mathsf{MOD}_0(m_0)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}) \neq \mathsf{MOD}_1(\mathsf{ADM})$, return $\perp$
      let $(m_i, \sigma_i) \leftarrow \mathsf{Sign}(1^\lambda, m_b, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{ADM})$
      $\underline{\text{let } (m'_0, \sigma'_0) \leftarrow \mathsf{Sanit}(1^\lambda, m_i, \mathsf{MOD}_0, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})}$
      $\underline{\text{let } (m'_1, \sigma'_1) \leftarrow \mathsf{Sanit}(1^\lambda, m_i, \mathsf{MOD}_1, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})}$
      $\underline{\text{if } |m'_0| \neq |m'_1|, \text{ return } \perp}$
      $\underline{\text{if } \forall 1 \leq i \leq |m'_0| : \exists i \text{ such that}}$
        $\underline{\mathsf{blkJudge}(1^\lambda, m'_0, \sigma'_0, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \perp, i) \neq \mathsf{blkJudge}(1^\lambda, m'_1, \sigma'_1, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \perp, i), \text{ return } \perp}$
      return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_i, \mathsf{MOD}_b, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$
  return 1, if $a = b$
```

Figure 84. Standard Privacy Experiment for SSS adjusted for block-level non-interactive public accountability; underlined content indicates differences compared to the game of the harmonised definition of Definition 156 (see Fig. 51 on page 265)

The additions force that for both of the adversary-provided inputs a run of blkJudge attributes all block of the oracle's output to the same origin. The same additions can be straightforwardly applied for schemes that offer group-level non-interactive public accountability. The adjusted game under grpJudge is omitted for brevity.

[797] An 'optimised' oracle could skip the final sanitization from the original game, but instead output the already computed sanitized message-signature pair that corresponds to the random coin toss.

13.4.3. *pubacc$\mathcal{SSS}$* algorithmic description

The additional algorithm blkJudge enables defining (weak) blockwise non-interactive public accountability as defined in Def. 190 if it works correctly even on an empty proof, i.e. $\pi = \bot$. The algorithm blkJudge was not part of the original SSS description by *Ateniese et al.*, since it is not required for the purpose of a standard SSS with message-level properties [12, 64]. However, it is required due to the definition of new properties, e.g., (weak) block-level non-interactive public accountability. The algorithm blkJudge is defined as follows:

Definition 195 : New Algorithm blkJudge for SSS.

blkJudge for SSS: *On input of the security parameter λ, a message-signature pair (m, σ), the proof, the corresponding public keys pk_{sig} and pk_{san}, the algorithm blkJudge returns a list of d_i, where $d_i \in \{\texttt{Sig}, \texttt{San}\}$ indicates which party is accountable for the block $m[i]$ (or $\bot$, indicating an error):*

$$d_i \leftarrow \mathsf{blkJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi, i)\,.$$

Note, the algorithm can allow for non-interactive public accountability of blocks when it works correctly on an empty proof, i.e., $\pi = \bot$.

Definition 196 : *pubacc$\mathcal{SSS}$*: Non-Interactive Public Accountable Sanitizable Signature Scheme

pubacc$\mathcal{SSS}$ which offers non-interactive public (PUB) accountability for messages consists of eight efficient (PPT) algorithms $(\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$, such that:

Key Generation: *There are two key generation algorithms, one for the signer and one for the sanitizer. Both create a pair of keys consisting of a private key and the corresponding public key:*

$$(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda), \qquad (\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda)\,.$$

The security parameter is denoted by 1^λ.

Signing: *The Sign algorithm takes as input the security parameter 1^λ, a message $m \in \{0,1\}^*$, the signer's secret key sk_{sig}, the sanitizer's public key pk_{san}, as well as a description ADM of the admissibly modifiable blocks. ADM is a set containing just those blocks' indices which are modifiable by the sanitizer. It outputs a signature σ and the message (or $\bot$ on error):*

$$(m, \sigma) \leftarrow \mathsf{Sign}(1^\lambda, m, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM})\,.$$

It is assumed that ADM and $m[\mathsf{fix}]$ are always recoverable from any signature $\sigma \neq \bot$ by the sanitizer holding the secret key sk_{san} that corresponds to pk_{san}.

Sanitizing: *Algorithm Sanit takes the security parameter, a message $m \in \{0,1\}^*$, a modification instruction MOD, a signature σ, the signer's public key pk_{sig} and the sanitizer's secret key sk_{san}. It modifies the message m according to the modification instruction MOD, which contains pairs $(i, m[i]')$ for those blocks that shall be modified. Sanit generates a new signature σ' for the modified message $m' \leftarrow \mathsf{MOD}(m)$. Then Sanit outputs m' and σ' (or $\bot$ in case of an error):*

$$(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})\,.$$

Verification: *The Verify algorithm outputs a decision $d \in \{\texttt{false}, \texttt{true}\}$ indicating the correctness of the signature σ for the message m with respect to the public keys pk_{sig} and pk_{san}. An output of* `true` *stands for a valid signature, while* `false` *indicates an invalid signature or an error:*

$$d \leftarrow \mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san})\,.$$

Proof: *The* Proof *algorithm takes as input the security parameter, the signer's secret key* sk_{sig}*, a message* m *and a signature* σ *as well a set of (polynomially many) additional message-signature pairs* $(m_i, \sigma_i)_{i=1,2,\ldots,k}$ *and the public key* pk_{san}*. It outputs a string* $\pi \in \{0,1\}^*$*:*

$$\pi \leftarrow \mathsf{Proof}(1^\lambda, \mathsf{sk}_{sig}, m, \sigma, (m_1, \sigma_1), \ldots, (m_k, \sigma_k), \mathsf{pk}_{san}) .$$

Judge / blkJudge: *Algorithm* Judge *takes as input the security parameter, a message* m *and a valid signature* σ*, the public keys of the parties and an empty proof* π*. It outputs a decision* $d \in \{\texttt{Sig}, \texttt{San}\}$ *indicating whether the message-signature pair has been created by the signer or the sanitizer (or* $\perp$ *in case of an error):*

$$d \leftarrow \mathsf{Judge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi) .$$

Respectively:
Algorithm blkJudge *takes as input the security parameter, a message* m *and a valid signature* σ*, the public keys of the parties and an empty proof* π*. It outputs a list of* d_i*, where* $d_i \in \{\texttt{Sig}, \texttt{San}\}$ *indicates which party is accountable for the block* $m[i]$ *(or* $\perp$ *in case of an error):*

$$d_i \leftarrow \mathsf{blkJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi, i) .$$

13.4.4. *pubaccSSS* correctness properties

For *pubaccSSS* the usual correctness properties as described in Sec. 11.4 for SSS are required to hold. In particular, every genuinely signed or sanitized message is accepted and genuinely created proofs by the signer lead the judge to decide in favour of the signer.

Further it holds also for the block-level judge (blkJudge). Finally, the correct behaviour of *pubaccSSS* includes that due to its non-interactive public accountability it will have an empty Proof algorithm. This requires that Judge and blkJudge always correctly decides upon an empty proof ($\pi = \perp$). This simplifies the description, as Proof in the interactive version required the Signer's private key $\mathsf{sk}_{\mathrm{sig}}$.

13.5. *pubaccSSS* cryptographic instantiation

The *pubaccSSS* scheme's cryptographic instantiation and the proofs given in this section have in part already been published at EuroPKI 2012 as joint work with *C. Brzuska* and *K. Samelin* [68] (see Appendix A publication n° 11).

13.5.1. *pubaccSSS* constructions

The *pubaccSSS* as well as the *unlinkableSSS* scheme are both built upon the same idea[798]. They generate two signatures (see Fig. 85): one to sign the fix part of m, i.e., $m[\mathsf{fix}]$, called the "inner signature", and another one to sign the admissible parts, i.e., $m[\mathsf{ADM}]$, together with the fixed parts, called the "outer signature". The inner signature is produced by the Signer of the signature scheme, while the outer signature can be produced by either one, the Signer or the Sanitizer. Using different signature types as inner and outer signature yields different properties of the sanitizable signature scheme. For instance, the existing work by *Brzuska et al.* [67] uses a group signature for the outer signature. The anonymity of the group signature makes signatures of the Signer and the Sanitizer indistinguishable, and the non-frameability/traceability property of the group signature scheme assures an interactive form of accountability. In the work by *Brzuska et al.* [65] the authors use standard signature schemes also for the outer signature. For *pubaccSSS* both signatures are standard CDSS.

[798] The following description in part is therefore also found in the results around *unlinkableSSS* published as joint work with *C. Brzuska* and *K. Samelin* [69] at EuroPKI 2013 (see Appendix A publication n° 17). The cryptographic instantiation and the proofs given in this section have in part already been published at EuroPKI 2012 as part of the joint work with *C. Brzuska* and *K. Samelin* [68] (see Appendix A publication n° 11).

Note, the *unlinkable$\mathcal{SSS}$* scheme uses a deterministic signature scheme as the inner signature and thereby the scheme satisfies both, a public, non-interactive version of accountability and a statistical notion of unlinkability, the strongest notion of privacy.

By dropping the need for transparency *pubacc$\mathcal{SSS}$* can drop non standard signatures and thereby gains efficiency. It is not transparent anymore, but it still enjoys privacy and a non-interactive public form of accountability, thus complying with legal standards. As transparency is sometimes seen as a stronger notion of privacy, one might feel that one has to compromise. One might fear that a strong notion of accountability would require to have a privacy protection weaker to standard privacy. That this is actually not the case can be shown as follows: As the inner signature scheme, one uses a deterministic signature scheme and proves that the scheme satisfies both, a public non-interactive version of accountability (PUB) and simultaneously a statistical notion of unlinkability, the strongest notion of privacy.

The use of the CDSS as building blocks for SSS maintains a high compatibility with existing legally recognised schemes. Especially, it allows to build upon existing public-key infrastructure for the linkage of the Signer's and Sanitizer's public key to the originator. This is helpful to achieve Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1). Further, this way of construction means that the resulting scheme only requires a constant number of standard cryptographic operations.

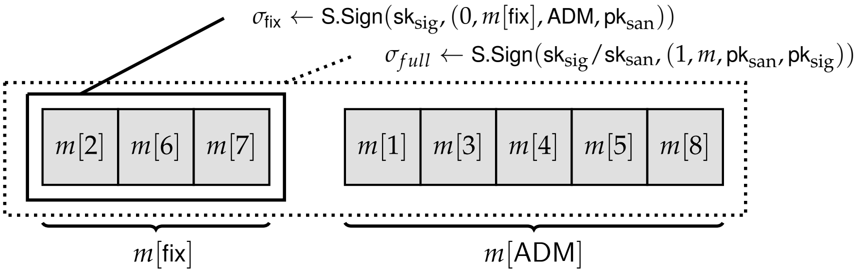

$\sigma_{\mathsf{fix}} \leftarrow \mathsf{S.Sign}(\mathsf{sk_{sig}}, (0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk_{san}}))$
$\sigma_{full} \leftarrow \mathsf{S.Sign}(\mathsf{sk_{sig}}/\mathsf{sk_{san}}, (1, m, \mathsf{pk_{san}}, \mathsf{pk_{sig}}))$
$m[2]$
$m[6]$
$m[7]$
$m[1]$
$m[3]$
$m[4]$
$m[5]$
$m[8]$
$m[\mathsf{fix}]$
$m[\mathsf{ADM}]$

Figure 85. Construction idea: Blocks 2,6,7 of m are fixed and together with $\mathsf{pk_{san}}$ and ADM signed by signer (σ_{fix}); the complete message m is signed by either the signer or the sanitizer (σ_{full}); as jointly published in *Brzuska, Pöhls, and Samelin* [69]; notation adjusted

In the following, two constructions are presented[799]: The first construction is non-interactive publicly (PUB) accountable, while the second one achieves block-level non-interactive public (PUB) accountability. Note, both constructions for *pubacc$\mathcal{SSS}$* still satisfy standard privacy, immutability and unforgeability.

13.5.1.1. Construction 1: Non-interactive public (PUB) accountability

Brzuska et al. [65] give a construction that achieves all properties defined in [64] except for transparency. Taking into account that accountability in their scheme works without the need of the interactive Judge algorithm, their construction is in line with Theorem 4 and thus their construction also features non-interactive public (PUB) accountability. However, *Brzuska et al.* [65] did neither formally nor explicitly define this property. The main construction idea is that the Signer signs the fixed parts of the message and additionally, the whole message is signed by the Signer or the Sanitizer, see Fig. 85 for an overview.

Construction 1 : *pubacc$\mathcal{SSS}$*

Let $\mathsf{S} = (\mathsf{S.KGen}, \mathsf{S.Sign}, \mathsf{S.Verify})$ be an unforgeable signature scheme. Define the sanitizable signature scheme pubacc$\mathcal{SSS}$ = $(\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Judge})$ as follows:

[799] The *pubacc$\mathcal{SSS}$* scheme's cryptographic instantiation have in part already been published at EuroPKI 2012 as joint work with *C. Brzuska* and *K. Samelin* [68] (see Appendix A publication nº 11).

Key Generation: *Algorithm KGen_{sig} generates on input of the security parameter 1^λ a key pair $(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{S.KGen}(1^\lambda)$ of the underlying signature scheme S, and algorithm KGen_{san} for input 1^λ analogously returns a pair $(\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{S.KGen}(1^\lambda)$.*

Signing: *Algorithm Sign on input of $m \in \{0,1\}^*, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM}$ computes*

$$\sigma_{\mathsf{fix}} \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{sig}, (0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san})),$$

$$\sigma_{\mathsf{full}} \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{sig}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}))$$

using the underlying signing algorithm. It returns $(m, \sigma) = (m, (\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM}))$.

Sanitizing: *Algorithm Sanit on input of the message m, modification instructions MOD, a signature $\sigma = (\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM})$, keys pk_{sig} and sk_{san} first checks that MOD is admissible according to ADM and that σ_{fix} is a valid signature for message $(0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san})$ under key pk_{sig}. If not, it stops outputting $\perp$. If no error occurred, it generates the modified message $m' \leftarrow \mathsf{MOD}(m)$ and computes*

$$\sigma'_{\mathsf{full}} \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{san}, (1, m', \mathsf{pk}_{san}, \mathsf{pk}_{sig}))$$

and outputs $(m', \sigma') = (m', (\sigma_{\mathsf{fix}}, \sigma'_{\mathsf{full}}, \mathsf{ADM}))$.

Verification: *Algorithm Verify on input of a message $m \in \{0,1\}^*$, a signature $\sigma = (\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM})$ and public keys pk_{sig}, pk_{san} first checks that σ_{fix} is a valid signature for message $(0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san})$ under key pk_{sig}, and $m[\mathsf{fix}]$ matches m under ADM. It returns `true`, if either $\mathsf{S.Verify}(\mathsf{pk}_{sig}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}), \sigma_{\mathsf{full}}) =$ `true` or $\mathsf{S.Verify}(\mathsf{pk}_{san}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}), \sigma_{\mathsf{full}}) =$ `true`, i.e. if either the signature verifies under pk_{sig} or pk_{san}. If so, Verify outputs `true`, declaring the entire signature as valid. Otherwise it returns `false`.*

Proof: *The Proof algorithm always returns $\perp$.*

Judge: *Judge on input of $m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}$ and $\perp$ parses σ as $(\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM})$ and outputs `Sig` if $\mathsf{S.Verify}(\mathsf{pk}_{sig}, (1, m, \mathsf{ADM}, \mathsf{pk}_{san}), \sigma_{\mathsf{full}}) =$ `true`, else if $\mathsf{S.Verify}(\mathsf{pk}_{san}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}) =$ `true` then it returns `San`. Note, at least one of these two verifies, because Judge is only run on valid pairs (m, σ).*

13.5.1.2. Construction 2: Block-level non-interactive public (PUB) accountability

The basic idea of the second construction is depicted in Fig. 86. It satisfies block-level non-interactive public (PUB) accountability.

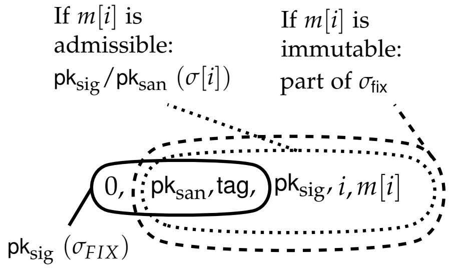

If $m[i]$ is admissible: $\mathsf{pk}_{\mathsf{sig}}/\mathsf{pk}_{\mathsf{san}}$ $(\sigma[i])$
If $m[i]$ is immutable: part of σ_{fix}
$0,$ $\mathsf{pk}_{\mathsf{san}}, \mathsf{tag},$ $\mathsf{pk}_{\mathsf{sig}}, i, m[i]$
$\mathsf{pk}_{\mathsf{sig}}$ (σ_{FIX})

Figure 86. Construction 2: Block-level public accountability due to the block-level signatures $\sigma[i]$ (A sanitized message additionally contains tag', which is not shown here for brevity); as jointly published in *Brzuska, Pöhls, and Samelin* [68]

Each message first gets a unique random message identifier tag to prevent mix and match attacks. The Signer then signs the fixed blocks together with tag. Additionally, it binds each of the modifiable blocks separately to the fixed message part and tag via a blockwise signature. To replace a block, the Sanitizer removes the existing blockwise signature and re-signs it using $\mathsf{sk}_{\mathrm{san}}$. However, a malicious Signer could re-use the tag and include blocks from previous messages which bear a Sanitizer's signature. To prevent this, the sanitizer additionally binds $m[i]'$ to a new random tag tag'. Note, in this second construction, $m[\mathsf{fix}]$ needs to part of the signature processing algorithms, binding each block to the same fixed blocks.

Construction 2 : $\boldsymbol{pubacc\mathcal{SSS}^{Block}}$

Let $S = (\mathsf{S.KGen}, \mathsf{S.Sign}, \mathsf{S.Verify})$ be a regular signature scheme. Define the sanitizable signature scheme $pubacc\mathcal{SSS}^{Block} = (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Judge}, \mathsf{blkJudge})$ as follows:

Key Generation: *Algorithm KGen_{sig} generates on input of the security parameter 1^λ a key pair $(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{S.KGen}(1^\lambda)$ of the underlying signature scheme, and algorithm KGen_{san} for input 1^λ analogously returns a pair $(\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{S.KGen}(1^\lambda)$.*

Signing: *Algorithm Sign on input of $m \in \{0,1\}^*, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM}$ draws a random tag tag and computes for all fixed blocks $m[\mathsf{fix}]$*

$$\sigma_{\mathsf{fix}} \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{sig}, (0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san}, \mathsf{tag}))$$

and for each block $m[i]$ with $i \in \mathsf{ADM}$, it computes

$$\sigma[i] \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{sig}, (i, m[i], m[\mathsf{fix}], \mathsf{pk}_{san}, \mathsf{pk}_{sig}, \mathsf{tag}, \bot)).$$

It returns $(m, \sigma) = (m, (\sigma_{\mathsf{fix}}, (\sigma[i])_{i \in \mathsf{ADM}}, \mathsf{ADM}, \mathsf{tag}, \bot))$.

Sanitizing: *Algorithm Sanit on input of a message m, modification instruction MOD, a signature $\sigma = (\sigma_{\mathsf{fix}}, (\sigma[i])_{i \in \mathsf{ADM}}, \mathsf{ADM}, \mathsf{tag})$, keys pk_{sig} and sk_{san} first verifies that the signature-message pair is valid by running Verify. If not (`false`), it stops and returns $\bot$. Otherwise it continues and generates $m' \leftarrow \mathsf{MOD}(m)$.*
If the changed m' does not conform to ADM, it stops and returns $\bot$. Otherwise it continues and draws a random tag tag' and for each $(i, m[i]') \in \mathsf{MOD}$ and computes

$$\sigma[i]' \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{san}, (i, m[i]', m[\mathsf{fix}], \mathsf{pk}_{san}, \mathsf{pk}_{sig}, \mathsf{tag}, \mathsf{tag}')) .$$

For each $(i, m[i]) \in \mathsf{ADM} \setminus \mathsf{MOD}$, it sets $\sigma[i]' := \sigma[i]$. It then outputs m' together with $\sigma' = (\sigma_{\mathsf{fix}}, (\sigma_i)'_{i \in \mathsf{ADM}}, \mathsf{ADM}, \mathsf{tag}, \mathsf{tag}')$.

Verification: *Algorithm Verify on of input a message $m \in \{0,1\}^*$, a signature $\sigma = (\sigma_{\mathsf{fix}}, (\sigma[i])_{i \in \mathsf{ADM}}, \mathsf{ADM}, \mathsf{tag}, \mathsf{tag}')$ and public keys pk_{sig}, pk_{san} the algorithm first checks that*
$\mathsf{S.Verify}(\mathsf{pk}_{sig}, (0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san}, \mathsf{tag}), \sigma_{\mathsf{fix}}) = $ `true`, hence σ_{fix} is a valid signature.
For each block $m[i]$ with $i \in \mathsf{ADM}$ it checks that at least one of the algorithms
$\mathsf{S.Verify}(\mathsf{pk}_{sig}, (i, m[i], m[\mathsf{fix}], \mathsf{pk}_{san}, \mathsf{pk}_{sig}, \mathsf{tag}, \bot), \sigma[i])$ or
$\mathsf{S.Verify}(\mathsf{pk}_{san}, (i, m[i], m[\mathsf{fix}], \mathsf{pk}_{san}, \mathsf{pk}_{sig}, \mathsf{tag}, \mathsf{tag}'), \sigma[i])$ returns `true`. If so, it outputs `true`. Otherwise it returns `false`, indicating an invalid signature.

Proof: *The Proof algorithm always returns $\bot$.*

Block-level Judge: *The blkJudge algorithm on input of $(m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, i)$ returns `Sig` if $\mathsf{S.Verify}(\mathsf{pk}_{sig}, (i, m[i], \mathsf{pk}_{san}, \mathsf{pk}_{sig}, \mathsf{tag}, \bot), \sigma[i]) = 1$ and `San` if $\mathsf{S.Verify}(\mathsf{pk}_{san}, (i, m[i]', m[\mathsf{fix}], \mathsf{pk}_{san}, \mathsf{pk}_{sig}, \mathsf{tag}, \mathsf{tag}'), \sigma[i]) = 1$. Note, at least one of the verifications succeeds, as blkJudge is only queried for valid pairs (m, σ).*

Judge: *The algorithm Judge on input of $m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}$ verifies using Verify that the signature is valid (`true`). If not, it aborts returning $\bot$. If the signature is valid, Judge returns `Sig`, if blkJudge marks **all** message blocks as `Sig`, otherwise `San`.*

13.5.2. $pubacc\mathcal{SSS}$ and $pubacc\mathcal{SSS}^{Block}$ complexity and runtime

Generating a signature in the message-level $pubacc\mathcal{SSS}$ requires exactly two signature generation steps, which is in $\mathcal{O}(1)$ in the number of blocks. For the block-level $pubacc\mathcal{SSS}^{Block}$ it requires an individual signature per admissible block and one σ_{FULL}. Still the required signature generation steps are only linear, i.e. $\mathcal{O}(n)$ where n is the amount of blocks. The same holds for the worst case of signature verification and for the sanitization operation.

To provide rough estimates, during the course of the joint publication [69] (see Appendix A publication n^{o} 17) both constructions have been implemented. This was done to demonstrate their practical usability. All tests were performed on an *Intel* T8300 Dual Core @2.40 GHz and 4 GiB of RAM, running *Ubuntu* Version 10.04 LTS (64 Bit) and Java version `1.6.0_26-b03`. Only a single thread was utilised to calculate the signatures; an improvement would be to parallelise signature calculations. The measurements given are the median of 100 runs. The signature is an RSA-PSS with a key size of 4096 Bit. To generate the timings both algorithms were run on two different block counts (10 and 100), the amount of admissible blocks got varied from 10% to a maximum of 90%, and for the measurements of the sanitization algorithm 10% of the admissible blocks were subject to sanitization (minimal 1 block). The results in Tab. 12 and Tab. 13 show that $pubacc\mathcal{SSS}$ and $pubacc\mathcal{SSS}^{Block}$ are reasonably performant with a high security parameter size and for high block counts: A maximum of 200 ms for signing with Construction 1 and less than 5 seconds for signing or sanitization with Construction 2.

	KeyGen	Sign		Sanit (10% of |ADM|)		Verify		Judge	
ℓ / |ADM|	10/100	10	100	10	100	10	100	10	100
10% of ℓ	18,649	200	200	101	101	9	13	11	13
90% of ℓ	18,649	165	165	102	98	8	14	11	16

Table 12. $pubacc\mathcal{SSS}$: Performance of message-level scheme from Construction 1; ℓ is the number of blocks; median runtime in ms; as jointly published in *Brzuska, Pöhls, and Samelin* [68]

	KeyGen	Sign		Sanit (10% of |ADM|)		Verify		blkJudge	
ℓ / |ADM|	10/100	10	100	10	100	10	100	10	100
10% of ℓ	18,649	565	4,569	110	932	24	126	38	301
90% of ℓ	18,649	547	4,164	443	4,266	27	187	53	385

Table 13. $pubacc\mathcal{SSS}^{Block}$: Performance of block-level scheme from Construction 2; ℓ is the number of blocks; median runtime in ms; as jointly published in *Brzuska, Pöhls, and Samelin* [68]

13.5.3. $pubacc\mathcal{SSS}$ and $pubacc\mathcal{SSS}^{Block}$ security proof

The following proofs of $pubacc\mathcal{SSS}$'s and $pubacc\mathcal{SSS}^{Block}$'s provable security can also be found in the related joint publication [68] (see Appendix A publication n^{o} 11).

A sanitization with an empty modification instruction must still make the Sanitizer accountable in order to maintain privacy while allowing non-interactive public accountability. It is required that MOD $= \emptyset$ indeed changes who is accountable. This got introduced as 'touching a message' in Definition 47 on page 49. As already discussed in Sec. 13.1.4 this difference is of paramount importance for non-interactive public accountability not to negatively impact on privacy. See Sec. 13.4.2 on page 329 for an adjusted privacy game and a discussion.

Theorem 13 : $pubacc\mathcal{SSS}$ from Construction 1 is Secure.

If the underlying signature scheme S is UNF-CMA, then the scheme $pubacc\mathcal{SSS}$ from Construction 1 is unforgeable, immutable, private, accountable and non-interactive publicly accountable.

Proof 13

The first four properties are proven by Brzuska et al. [65], and thus, by Theorem 4, Construction 1 is also non-interactive publicly accountable, as Proof returns $\perp$. □

Theorem 14 : $pubacc\mathcal{SSS}^{Block}$ from Construction 2 is Secure.

If the underlying signature scheme S is UNF-CMA, then the scheme $pubacc\mathcal{SSS}^{Block}$ from Construction 2 is unforgeable, immutable, private, accountable and block-level non-interactive publicly (PUB) accountable.

Proof 14

Block-level non-interactive public accountability implies non-interactive public (PUB) accountability and accountability; the latter implies unforgeability [64]. It is thus sufficient to prove privacy, immutability and blockwise non-interactive public accountability. Privacy holds information-theoretically, as the original content of the modified message blocks is not used by the algorithm. The proof for immutability can be done analogously as for Construction 1, only the additional tag needs to be taken into account. However, tag does not affect the analysis. It remains to prove block-level non-interactive public accountability.

Assume, there is an efficient adversary $\mathcal{A}$ against block-level non-interactive public accountability. Then an efficient adversary $\mathcal{B}$ against the UNF-CMA of the underlying signature scheme is constructed as follows. The adversary $\mathcal{B}$ gets as input a public key pk and flips a coin b. If $b = 0$, it sets $\mathsf{pk}_{sig} := \mathsf{pk}$ and runs $\mathsf{S.KGen}$ to generate $(\mathsf{pk}_{san}, \mathsf{sk}_{san})$. If $b = 1$, it sets $\mathsf{pk}_{san} := \mathsf{pk}$ and runs $\mathsf{S.KGen}$ to generate $(\mathsf{pk}_{sig}, \mathsf{sk}_{sig})$. Subsequently, to simulate the oracles for $\mathcal{A}$, the algorithm $\mathcal{B}$ runs the algorithms Sign and Sanit according to the specification with the exception that whenever a signature is generated under the secret key sk corresponding to pk, the adversary $\mathcal{B}$ does not generate the signature itself, but instead, it queries its signing oracle. In the end, the adversary $\mathcal{A}$ outputs a triple $(\mathsf{pk}^*, m^*, \sigma^*)$, from which $\mathcal{B}$ extract a valid signature as follows – one can distinguish between two cases, a malicious $\mathsf{Sanitizer}$ attack and a malicious Signer attack. The probability that the simulation was done for the wrong case, is exactly $\frac{1}{2}$.

Malicious Sanitizer.

- *I) The tag tag^* in $(m^*, \sigma^*, \mathsf{pk}^*)$ has never been returned by the signing oracle. $\mathcal{B}$ returns $(0, m[\mathsf{fix}]^*, \mathsf{ADM}^*, \mathsf{pk}^*_{san}, \mathsf{tag}^*, \perp)$ together with σ^*_{fix} as its forgery.*
- *II) The tag tag^* used in $(m^*, \sigma^*, \mathsf{pk}^*)$ has been returned by the signing oracle in the ith query. If $\mathsf{pk}^*_{san} \neq \mathsf{pk}_{san,i}$ then $(0, m[\mathsf{fix}]^*, \mathsf{ADM}^*, \mathsf{pk}^*_{san}, \mathsf{tag}^*, \perp)$ together with σ^*_{fix} is output. Else, there is a q with $m_i[q] \neq m^*[q]$, but the local signature $\sigma^*[q]$ verifies under the signers public key pk_{sig}. The adversary $\mathcal{B}$ returns $\sigma^*[q]$ together with $(q, m^*[q], m[\mathsf{fix}]^*, \mathsf{pk}^*_{san}, \mathsf{pk}_{sig}, \mathsf{tag}^*, \perp)$.*

Malicious Signer.

- *I) The tag tag'^* used in $(m^*, \sigma^*, \mathsf{pk}^*)$ has never been returned by the sanitizing oracle. Then, $\mathcal{B}$ searches for a q where the local signature $\sigma^*[q]$ verifies under the sanitizer's public key pk_{san} and returns $\sigma^*[q]$ together with $(q, m^*[q], m[\mathsf{fix}]^*, \mathsf{pk}^*_{san}, \mathsf{pk}_{sig}, \mathsf{tag}^*, \mathsf{tag}'^*)$.*
- *II) The tag tag'^* used in $(m^*, \sigma^*, \mathsf{pk}^*)$ has been returned by the sanitizing oracle in the jth query. If $\mathsf{pk}^*_{sig} \neq \mathsf{pk}_{sig,j}$ then $(q, m^*[q], m[\mathsf{fix}]^*, \mathsf{pk}^*_{san}, \mathsf{pk}_{sig}, \mathsf{tag}^*, \mathsf{tag}'^*)$ together with $\sigma^*[q]$ is output, for a q, where $\sigma^*[q]$ is a valid signature under pk_{san}. Else if $\mathsf{pk}^*_{sig} = \mathsf{pk}_{sig,j}$, then there is a q with $m'_j[q] \neq m^*[q]$, but the local signature $\sigma^*[q]$ verifies under the sanitizer's public key pk_{sig}. The adversary $\mathcal{B}$ returns $\sigma^*[q]$ together with $(q, m^*[q], m[\mathsf{fix}]^*, \mathsf{pk}^*_{san}, \mathsf{pk}_{sig}, \mathsf{tag}'^*)$.*

Analysis. *The loss in success probability that might occur via tag collisions is omitted, as $\mathsf{tag} \in \{0,1\}^{\lambda}$. The adversary $\mathcal{B}$ has then a success probability which is $\frac{1}{2}$ the success probability of $\mathcal{A}$. If $\mathcal{B}$ guesses correctly whether it should embed the key for a malicious sanitizer or a malicious signer attack, then $\mathcal{B}$ is successful, whenever $\mathcal{A}$ is, because the messages output by $\mathcal{B}$ were not queried by $\mathcal{B}$ to its signing oracle before. In detail, if the tag has never been returned by the oracle, then the message is clearly fresh. Else, if the tag has been output to $\mathcal{A}$, then it has not been signed together with the corresponding block. Hence, the message is fresh with respect to the block and thus also a successful forgery.* □

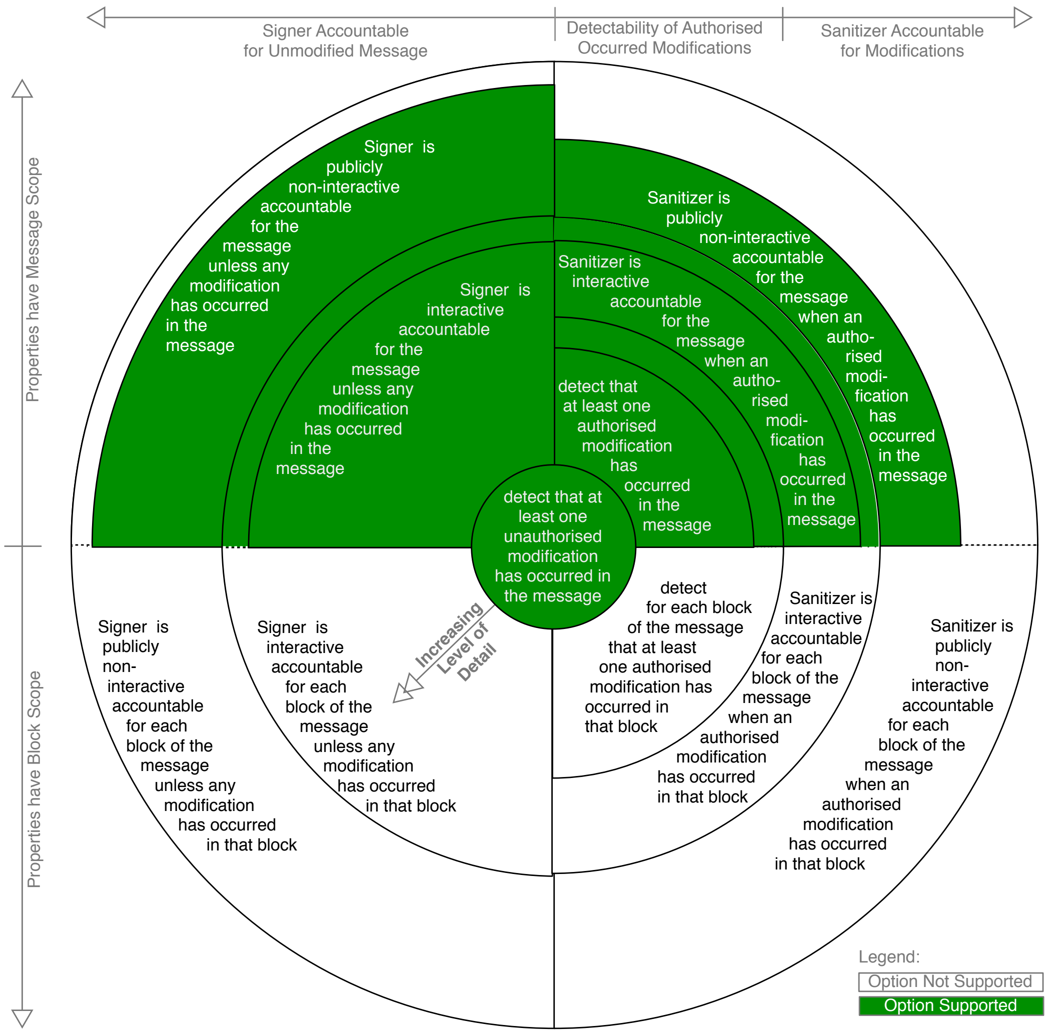

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Increasing Level of Detail
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 87. $pubacc\mathcal{SSS}$ offers ACA – 1CD – PUB integrity, because Verifiers can public non-interactively detect on the level of the message if an authorized or an unauthorized subsequent modification has been done or if the message is still original; $blockacc\text{-}\mathcal{SSS}$ only offers ACA – ≥1CD,< BCD – PUB integrity, because the Verifier detects only in which group of blocks the modification has happened (< BCD); additionally both offer accountability for Sanitizers once a block is sanitized there is accountability for Sanitizers for the full message once a block is sanitized

13.5.4. $pubacc\mathcal{SSS}$ and $pubacc\mathcal{SSS}^{Block}$ integrity offer and visualisation

In the following the integrity offers for the message-level scheme $pubacc\mathcal{SSS}$ and then for the block-level scheme $pubacc\mathcal{SSS}^{Block}$ are given.

13.5.4.1. $pubacc\mathcal{SSS}$ integrity offer and visualisation

$pubacc\mathcal{SSS}$ is a sanitizable signature scheme for an ordered list, e.g. an array. It offers the legally advised public form of accountability, i.e. non-interactive public accountability (PUB). Summarised, $pubacc\mathcal{SSS}$ classifies as giving ACA – ≥1CD– PUB integrity. The Signer stays accountable for the whole message until any subsequent modification happens. If it was an authorized modification then the change results in the Sanitizer becoming accountable for the message. The Verifier only needs public information in order to provide third party verifiable proofs of accountability. As accountability is done on the level of messages it achieves all levels on the upper half in the visualisation introduced in Sec. 6.9. This is depicted in Fig. 87.

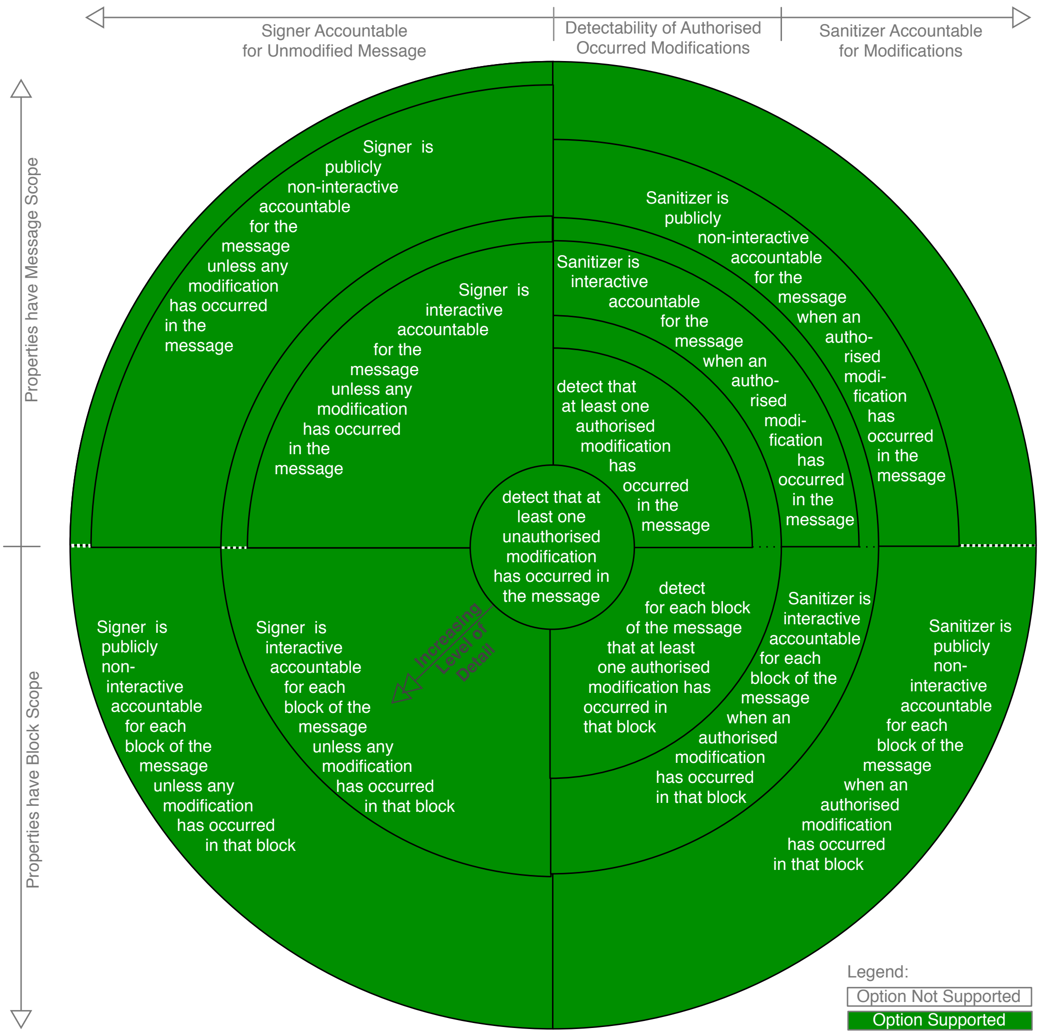

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Increasing Level of Detail
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 88. $pubacc\mathcal{SSS}^{Block}$ offers ACA – BCD – PUB integrity, because Verifiers can public non-interactively detect (PUB) on the level of blocks if an authorized or an unauthorized subsequent modification has occurred (≥1CD) and if a modification has happened the Verifier detects in which block (BCD)

13.5.4.2. $pubacc\mathcal{SSS}^{Block}$ integrity offer and visualisation

$pubacc\mathcal{SSS}^{Block}$ is a sanitizable signature scheme for an ordered list, e.g. an array. It offers the legally advised public form of accountability, i.e. non-interactive public accountability (PUB) for the level of block, i.e., a Verifier can identify for each single block if it has been modified (original, modified with authorization, modified without authorization). Summarised, $pubacc\mathcal{SSS}^{Block}$ classifies as giving ACA – BCD – PUB integrity. The Signer stays accountable for the whole message and each single block until any subsequent modification happens. If there was an authorized modification then the change results in the Sanitizer becoming accountable for all modified block(s), while the Signer still stays accountable for the blocks that were not modified. The Signer is no longer accountable for the whole message after the first successful sanitization of a block, if one assumes the intuitive construction of message-level accountability as an aggregation of all block's block-level property that was given in Definition 185 (see Sec. 13.1.5 on page 315). The Verifier only needs public information in order to provide third party verifiable proofs of accountability. As accountability is done on the level of blocks it achieves all levels that are visualisable in the visualisation introduced in Sec. 6.9. This is depicted in Fig. 88 on page 339.

13.6. $blockacc\mathcal{SSS}$ Concept: Non-Interactive public accountability for groups of block(s)

In the following the scheme $blockacc\mathcal{SSS}$ with non-interactive public accountability for grouped blocks is presented [800].

13.6.1. $blockacc\mathcal{SSS}$ extended security properties

The secure sanitizable signature scheme $blockacc\mathcal{SSS}$ achieves the following security properties (**bold** indicates a new security property achieved by this scheme):

- Unforgeability according to Definition 152, and
- Immutability according to Definition 154, and
- Standard Privacy following *Brzuska et al.* according to Definition 156, and

either **Group-of-Block-Level Public Non-Interactive Accountability (PUB)** according to Definition 190,

or **Group-of-Block-Level Transparency (INT)** according to Definition 191.

13.6.2. $blockacc\mathcal{SSS}$ algorithmic description

The algorithm grpJudge is newly introduced to complete $blockacc\mathcal{SSS}$. It was not part of the original SSS description by *Ateniese et al.*, since it is not required for the purpose of a standard SSS with message-level properties [12, 64]. However, it is required here due to the definition of a new accountability property on the level of grouped block.

Definition 197 : $blockacc\mathcal{SSS}$: Group-of-blocks-level Accountable Sanitizable Signature Scheme

$blockacc\mathcal{SSS}$ consists of at least the following efficient (PPT) algorithms $blockacc\mathcal{SSS} := (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge}, \mathsf{grpJudge})$:

Key Generation: *There are two key generation algorithms, one for the signer and one for the sanitizer. Both create a pair of keys consisting of a private key and the corresponding public key, based on the security parameter λ:*

$$(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda),$$

$$(\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda).$$

Signing: *The Sign algorithm takes as input the security parameter λ, a message $m = (m[1], \ldots, m[\ell])$, $m[i] \in \{0,1\}^*$, the secret key sk_{sig} of the signer, the public key pk_{san} of the sanitizer, as well as ADM and grp. It outputs the message m and a signature σ (or $\perp$, indicating an error):*

$$(m, \sigma) \leftarrow \mathsf{Sign}(1^\lambda, m, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM}, \mathsf{grp}).$$

Sanitizing: *Algorithm Sanit takes the security parameter λ, a message $m = (m[1], \ldots, m[\ell])$, $m[i] \in \{0,1\}^*$, a modification instruction MOD, a signature σ, the public key pk_{sig} of the signer and the secret key sk_{san} of the sanitizer. It modifies the message m according to the modification instruction MOD. Henceforth MOD is modelled to contain a list of pairs $(i, m[i]')$, indicating that block i shall be modified into the string $m[i]'$. Note, MOD can be empty or the string $m[i]'$ can be equal to $m[i]$. This allows the sanitizer to take accountability for a given group without modifying it. For simplicity, we write $\mathsf{grp}[j] \in \mathsf{MOD}$, if at least one block $j \in \mathsf{grp}[i]$ is to be modified. Sanit generates*

[800] The $blockacc\mathcal{SSS}$ scheme and the results given in this section have in part already been published at ARES 2013 as joint work with *K. Samelin, H. de Meer* and *J. Posegga* [132] (see Appendix A publication n° 16).

a new signature σ' for the modified message $m' = \mathsf{MOD}(m)$. Then Sanit *outputs m' and σ' (or $\perp$ in case of an error):*

$$(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san}) .$$

Verification: *The* Verify *algorithm outputs a decision $d \in \{\texttt{true}, \texttt{false}\}$, indicating the correctness of a signature σ for a message m with respect to the public keys pk_{sig} and pk_{san}:*

$$d \leftarrow \mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) .$$

Proof: *The* Proof *algorithm takes as input the security parameter λ, the secret signing key sk_{sig}, a message $m = (m[1], \ldots, m[\ell])$, $m[i] \in \{0,1\}^*$ and a signature σ as well a set of (polynomially many) additional message-signature pairs $\{(m_i, \sigma_i) | i \in \mathbb{N}\}$ and the public key pk_{san}. It outputs a string $\pi \in \{0,1\}^*$ (or $\perp$ in case of an error):*

$$\pi \leftarrow \mathsf{Proof}(1^\lambda, \mathsf{sk}_{sig}, m, \sigma, \{(m_i, \sigma_i) | i \in \mathbb{N}\}, \mathsf{pk}_{san}) .$$

Judge: *Algorithm* Judge *takes as input a message $m = (m[1], \ldots, m[\ell])$, $m[i] \in \{0,1\}^*$ and a valid signature σ, the public keys of the parties and a proof π. It outputs a decision $d \in \{\texttt{Sig}, \texttt{San}, \perp\}$, indicating whether the message-signature pair has been created by the signer or the sanitizer (or $\perp$ in case of an error):*

$$d \leftarrow \mathsf{Judge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi) .$$

Group-level Judge: *Algorithm* grpJudge *takes as input a message $m = (m[1], \ldots, m[\ell])$ with $m[i] \in \{0,1\}^*$, a group identifier g, a valid signature σ, the public keys of the parties and a proof π. It outputs a decision $d \in \{\texttt{Sig}, \texttt{San}, \perp\}$, indicating whether the respective group of blocks identified by g from the given message-signature pair has been created by the signer or the sanitizer (or $\perp$ in case of an error):*

$$d \leftarrow \mathsf{grpJudge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi, g) .$$

13.6.3. *blockacc$\mathcal{SSS}$* correctness properties

For *blockacc$\mathcal{SSS}$* the usual correctness properties as described in Sec. 11.4 for SSS are required to hold. In particular, every genuinely signed or sanitized message verifies as valid. Moreover, every genuinely created proof makes the judge decide in favour of the signer. For a non-interactively publicly accountable scheme this must also hold when the proof is empty, i.e. $\pi = \perp$. Further it holds also for the group-level judge (grpJudge).

13.7. *blockacc$\mathcal{SSS}$* cryptographic instantiation

The *blockacc$\mathcal{SSS}$* scheme and the results given in this section have in part already been published at ARES 2013 as joint work with *K. Samelin*, *H. de Meer* and *J. Posegga* [132] (see Appendix A publication n° 16). The first construction for *blockacc$\mathcal{SSS}$* is Construction 3 and it achieves group-level accountability with transparency (INT). The second construction, Construction 4, achieves group-level non-interactive public (PUB) accountability. Due to the relaxation of not needing transparency it requiring only a constant number of signatures and is thus more efficient than the transparent Construction 3.

13.7.1. *blockacc$\mathcal{SSS}$* cryptographic preliminaries

Both constructions (Construction 3 and 4) make use of the tag-based chameleon hash-function given by *Brzuska et al.* [64]. In particular, the chameleon hash-function must be collision-resistant under random tagging-attacks as assumed and shown in [64].

```
Experiment $\mathsf{Rand} - \mathsf{Tag}^{\mathcal{CH}}_{\mathcal{A}}(\lambda)$
  $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{CHKeyGen}(1^\lambda)$
  $(\mathsf{TAG}, m, r, \mathsf{TAG}', m', r') \leftarrow \mathcal{A}^{\mathsf{OAdapt}(\mathsf{sk},\cdot,\cdot,\cdot,\cdot)}(\mathsf{pk})$
    where oracle OAdapt for the $i^{\text{th}}$ query
    $(\mathsf{TAG}_i, m_i, r_i, m_i')$ with $\mathsf{TAG}_i \in \{0,1\}^\lambda$
    let $\mathsf{TAG}_i' \leftarrow \{0,1\}^\lambda$ and compute
    $r_i' \leftarrow \mathsf{CHAdapt}(\mathsf{sk}, \mathsf{TAG}_i, m_i, r_i, \mathsf{TAG}_i', m_i')$
    return $(\mathsf{TAG}_i', r_i')$
  return 1, if
    $(\mathsf{TAG}, m) \neq (\mathsf{TAG}', m')$ and
    let $i = 1, \ldots, q$ denote the $i^{\text{th}}$ oracle query
    $\mathsf{CHash}(\mathsf{pk}, \mathsf{TAG}, m, r) = \mathsf{CHash}(\mathsf{pk}, \mathsf{TAG}', m', r')$ and
    $\forall i, j : \{(\mathsf{TAG}, m), (\mathsf{TAG}', m')\} \neq \{(\mathsf{TAG}_i, m_i), (\mathsf{TAG}_i', m_i')\}$
    $\wedge \{(\mathsf{TAG}, m), (\mathsf{TAG}', m')\} \neq \{(\mathsf{TAG}_i', m_i'), (\mathsf{TAG}_j', m_j')\}$
```

Figure 89. Collision-Resistance against Random Tagging Attacks Experiment [64]

13.7.1.1. Chameleon hash-function with tags ($\mathcal{CH}$)

Definition 198 : Chameleon Hash-Function with Tags $\mathcal{CH}$

A chameleon hash-function $\mathcal{CH} := (\mathit{CHKeyGen}, \mathit{CHash}, \mathit{CHAdapt})$ with tags consists of three efficient (PPT) algorithms:

CHKeyGen: The algorithm CHKeyGen takes as input the security parameter 1^λ and outputs the key pair required for the chameleon hash-function:

$$(\mathit{sk}, \mathit{pk}) \leftarrow \mathit{CHKeyGen}(1^\lambda)\,.$$

CHash: The algorithm CHash takes as input the public key pk, a string m to hash, a tag TAG and a random number $r \in \{0,1\}^\lambda$. It outputs the digest h:

$$h \leftarrow \mathit{CHash}(1^\lambda, \mathit{pk}, \mathsf{TAG}, m, r)\,.$$

CHAdapt: The algorithm CHAdapt takes as input the private key sk, m, m', TAG, TAG$'$, r. It outputs the new random number r':

$$r' \leftarrow \mathit{CHAdapt}(1^\lambda, \mathit{sk}, \mathsf{TAG}, m, r, \mathsf{TAG}', m')\,.$$

Just for differentiation to increase readability, this thesis calls the output of the algorithm CHash 'digest', rather than hash-value. For the chameleon hash-function with tags $\mathcal{CH}$ all correctness properties are required to hold. In particular, $\mathsf{CHash}(\mathsf{pk}, \mathsf{TAG}, m, r) = \mathsf{CHash}(\mathsf{pk}, \mathsf{TAG}', m', r')$ must yield, if r' has been generated genuinely using CHAdapt.

Definition 199 : Collision-Resistance vs. Random-Tag Attacks from [64]

A tag-based chameleon hash-function $\mathcal{CH}$ is said to be collision-resistant under random-tagging attacks, if the probability that the experiment depicted in Fig. 89 returns 1 is negligible (as a function of λ). [64]

A concrete secure instantiation is found in [64]. Note, the distribution of r' is computationally indistinguishable from uniform [64].

13.7.2. $\mathit{blockacc}\mathcal{SSS}$ constructions

Construction 3 introduces a provably secure[801] construction which is transparent, private, immutable, group-level accountable and unforgeable. Construction 4 presents a provably secureconstruction which is private, immutable, group-level non-interactive publicly accountable (PUB) and unforgeable based on Construction 3. Construction 4 ($\mathit{blockacc}\mathcal{SSS}^{PUB}$) alters Construction 3 ($\mathit{blockacc}\mathcal{SSS}^{INT}$) such that it removes transparency, but efficiently gives group-level non-interactive public accountability. Construction 4 even achieves this with a constant number of signatures, which is more efficient than $\mathit{blockacc}\mathcal{SSS}^{INT}$ from Construction 3 where the number of signatures increased linearly with the number of blocks.

13.7.2.1. Construction 3: Group-level interactive non-public (INT) accountable SSS

Construction 3 uses the ideas from [64, 216]. In particular, each group is hashed using a tag-based chameleon hash-function. However, instead of using one tag for the complete message m, the construction uses different tags for each group $\mathsf{grp}[i]$. Construction 3 utilises a standard UNF-CMA signature scheme to generate the final signature. It also requires a pseudorandom function $\mathcal{PRF}$ and a pseudorandom generator $\mathcal{PRG}$.

Construction 3 : $\mathit{blockacc}\mathcal{SSS}^{INT}$

Let $\mathcal{SS} = (\mathsf{SKeyGen}, \mathsf{SSign}, \mathsf{SVerify})$ be standard UNF-CMA secure signature scheme. Let $\mathcal{PRF}$ be a pseudorandom function mapping n-bit input on a n-bit output for n-bit keys. Let $\mathcal{PRG}$ be a pseudorandom generator mapping n-bit inputs to $2n$-bit outputs. Define $\mathit{blockacc}\mathcal{SSS} = (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge}, \mathsf{grpJudge})$ as follows:

Key Generation KGen_{sig}: *Generate a key pair of the underlying signature algorithm $\mathsf{SKeyGen}$, i.e., $(pk, sk) \leftarrow \mathsf{SKeyGen}(1^\lambda)$. Pick a key $\kappa \leftarrow \{0,1\}^\lambda$ for the $\mathcal{PRF}$. Output $(pk_{sig}, sk_{sig}) = (pk, (sk, \kappa))$.*

Key Generation KGen_{san}: *Generate a key pair of the underlying chameleon hash-function. Output $(pk_{san}, sk_{san}) \leftarrow \mathsf{CHKeyGen}(1^\lambda)$.*

Signing: *On input of $m = (m[1], \ldots, m[\ell])$, $m[i] \in \{0,1\}^*$, pk_{san}, sk_{sig}, ADM, and grp, Sign draw $\gamma + 1$ nonces $n_i \leftarrow \{0,1\}^\lambda$ and compute: $x_i \leftarrow \mathcal{PRF}(\kappa, n_i)$ and $\mathsf{TAG}_i \leftarrow \mathcal{PRG}(x_i)$ for all $i = 0, \ldots, \gamma$. Draw $\gamma + 1$ additional nonces $r_i \leftarrow \{0,1\}^\lambda$. Let:*

$$h[i] \leftarrow \mathsf{CHash}(pk_{san}, \mathsf{TAG}_i, (i, \mathsf{grp}[i], pk_{sig}), r_i)$$

for all $i = 1, \ldots, \gamma$. Now, let:

$$h[0] \leftarrow \mathsf{CHash}(pk_{san}, \mathsf{TAG}_0, (\mathsf{TAG}_1, \ldots, \mathsf{TAG}_\gamma, m, pk_{sig}), r_0)\,.$$

Set

$$\sigma_c \leftarrow \mathsf{SSign}(sk, (h[0], \ldots, h[\gamma], \mathsf{grp}[0], pk_{san}, \mathsf{ADM}, \mathsf{grp}))\,.$$

Output (m, σ), where

$$\sigma = (\sigma_c, (\mathsf{TAG}_i)_{0 \le i \le \gamma}, (n_i)_{0 \le i \le \gamma}, \mathsf{ADM}, \mathsf{grp}, (r_i)_{0 \le i \le \gamma})\,.$$

Verification: *On input of pk_{sig}, pk_{san}, m and*

$$\sigma = (\sigma_c, (\mathsf{TAG}_i)_{0 \le i \le \gamma}, (n_i)_{0 \le i \le \gamma}, \mathsf{ADM}, \mathsf{grp}, (r_i)_{0 \le i \le \gamma})$$

compute for each $i \in \mathsf{grp}$:

$$h[i] \leftarrow \mathsf{CHash}(pk_{san}, \mathsf{TAG}_i, (i, \mathsf{grp}[i], pk_{sig}), r_i)$$

[801] As defined in Sec. 2.3.1.

and

$$h[0] \leftarrow \mathsf{CHash}(pk_{san}, \mathsf{TAG}_0, (\mathsf{TAG}_1, \dots, \mathsf{TAG}_\gamma, m), r_0)\,.$$

Output:

$$\mathsf{SVerify}(pk, (h[0], \dots, h[\gamma], grp[0], pk_{san}, ADM, grp), \sigma_c)\,.$$

Sanitize: *On input of pk_{sig}, sk_{san}, m, MOD and σ, first check, if the received message-signature pair is valid using Verify. Check, if $MOD \subseteq ADM$. If not, stop outputting $\perp$. For each* **group** *$grp[i] \in MOD$, draw a nonce $n_i' \leftarrow \{0,1\}^\lambda$ and a new tag $\mathsf{TAG}_i' \leftarrow \{0,1\}^{2\lambda}$. If $grp[i] \notin MOD$, tags, random values and nonces are copied from the original signature, i.e., $n_i' = n_i$ and $\mathsf{TAG}_i' = \mathsf{TAG}_i$. If $MOD \neq \emptyset$, draw an additional nonce $n_0' \leftarrow \{0,1\}^\lambda$ and an additional tag: $\mathsf{TAG}_0' \leftarrow \{0,1\}^{2\lambda}$. Compute:*

$$r_i' \leftarrow \mathsf{CHAdapt}(sk_{san}, \mathsf{TAG}_i, (i, grp[i], pk_{sig}), r_i, \mathsf{TAG}_i', (i, grp[i]', pk_{sig}))$$

for each $grp[i] \in MOD$ and

$$r_0' \leftarrow \mathsf{CHAdapt}(sk_{san}, \mathsf{TAG}_0, (\mathsf{TAG}_1, \dots, \mathsf{TAG}_\gamma, m),$$
$$r_0, \mathsf{TAG}_0', (\mathsf{TAG}_1', \dots, \mathsf{TAG}_\gamma', m'))\,.$$

Output (m', σ'), where $m' \leftarrow MOD(m)$ and

$$\sigma' = (\sigma_c, (\mathsf{TAG}')_{0 \le i \le \gamma}, (n_i')_{0 \le i \le \gamma}, ADM, grp, (r_i')_{0 \le i \le \gamma})\,.$$

Proof: *On input of sk_{sig}, m, $\sigma = (\sigma_c, (\mathsf{TAG}_i)_{0 \le i \le \gamma}, (n_i)_{0 \le i \le \gamma}, ADM, grp, (r_i)_{0 \le i \le \gamma})$, pk_{san} and a sequence of message-signature pairs $\{(m_i, \sigma_i) \mid i \in \mathbb{N}\}$, search for all groups the matching signatures, such that:*

$$\mathsf{CHash}(pk_{san}, \mathsf{TAG}_i, (i, grp[i], pk_{sig}), r_i) =$$
$$\mathsf{CHash}(pk_{san}, \mathsf{TAG}_i', (i, grp'[i], pk_{sig}), r_i')\,.$$

Do the same for the outer chameleon hash-value:

$$\mathsf{CHash}(pk_{san}, \mathsf{TAG}_0, (\mathsf{TAG}_1, \dots, \mathsf{TAG}_\gamma, m), r_i) =$$
$$\mathsf{CHash}(pk_{san}, \mathsf{TAG}_0', (\mathsf{TAG}_1', \dots, \mathsf{TAG}_\gamma', m'), r_i')\,.$$

Set $TAG_i \leftarrow \mathcal{PRG}(x_i)$, where $x_i \leftarrow \mathcal{PRF}(\kappa, n_i)$. Output π, where

$$\pi = ((\mathsf{TAG}_i)_{0 \le i \le \gamma}, m, pk_{sig}, pk_{san}, (r_i)_{0 \le i \le \gamma}, (x_i)_{0 \le i \le \gamma})\,.$$

If any errors occur, output $\perp$. In other words, Proof outputs the original blocks as the proof for the complete message.

grpJudge: *On input of m, σ, pk_{sig}, pk_{san}, an index i, and the proof*

$$\pi = ((\mathsf{TAG}_i^\pi)_{0 \le i \le \gamma}, m^\pi, pk_{sig}^\pi, pk_{san}^\pi, (r_i^\pi)_{0 \le i \le \gamma}, (x_i^\pi)_{0 \le i \le \gamma}).$$

First, check if σ verifies. Afterwards, check, if $pk_{san}^\pi = pk_{san}$. Else, return $\perp$. Let

$$d_i \leftarrow \begin{cases} \mathtt{San} & \text{if the collision is non-trivial and} \\ & \mathsf{TAG}_i^\pi = \mathcal{PRG}(x_i^\pi) \\ \mathtt{Sig} & \text{else} \end{cases}$$

If $\mathsf{TAG}_0^\pi \neq \mathcal{PRG}(x_0^\pi)$ and there exists no non-trivial collision for the outer chameleon-hash, set $d_i = \mathtt{Sig}$. Output d_i, or $\perp$ on error.

Judge: *On input of m, σ, pk_{sig}, pk_{san} and the proof*

$$\pi = ((\mathsf{TAG}_i^\pi)_{0 \le i \le \gamma}, m^\pi, pk_{sig}^\pi, pk_{san}^\pi, (r_i^\pi)_{0 \le i \le \gamma}, (x_i^\pi)_{0 \le i \le \gamma})$$

let $d_i \leftarrow$ grpJudge$(m, \sigma, pk_{sig}, pk_{san}, \pi, i)$ for each group grp$[i] \in$ grp. If $\mathsf{TAG}_0^{\pi} \neq \mathcal{PRG}(x_0^{\pi})$ and there exists no non-trivial collision for the outer chameleon-hash, output `Sig`. *On error, output $\perp$. If $\exists i : d_i \neq$* `Sig`, *then output* `San` *and* `Sig` *otherwise.*

13.7.2.2. Construction 4: Group-level non-interactive public (PUB) accountable SSS

Construction 4 : $blockacc\mathcal{SSS}^{PUB}$

Let $\mathcal{SS} = (\mathsf{SKeyGen}, \mathsf{SSign}, \mathsf{SVerify})$ be standard UNF-CMA secure signature scheme. Let $\mathcal{PRF}$ be a pseudorandom function mapping n-bit input on a n-bit output for n-bit keys. Let $\mathcal{PRG}$ be a pseudorandom generator mapping n-bit inputs to $2n$-bit outputs. Define blockacc$\mathcal{SSS}$ = (KGen$_{sig}$, KGen$_{san}$, Sign, Sanit, Verify, Proof, Judge) as follows:

Key Generation KGen$_{sig}$: *Generate a key pair of the underlying signature algorithm* SKeyGen, *i.e.,* $(pk, sk) \leftarrow \mathsf{SKeyGen}(1^{\lambda})$. *Output* $(pk_{sig}, sk_{sig}) = (pk, sk)$.

Key Generation KGen$_{san}$: *Generate two key pairs, one for the underlying chameleon hash-function and one for an unforgeable signature scheme. In particular, let* $(pk_{san}.pk_c, sk_{san}.sk_c) \leftarrow CHKeyGen(1^{\lambda})$ *and* $(pk_{san}.pk_s, sk_{san}.sk_s) \leftarrow \mathsf{SKeyGen}(1^{\lambda})$. *Output* $(pk_{san}, sk_{san}) = ((pk_{san}.pk_c, pk_{san}.pk_s), (sk_{san}.sk_c, sk_{san}.sk_s))$.

Signing: *On input of the message $m = (m[1], \ldots, m[\ell])$, $m[i] \in \{0,1\}^*$, pk_{san}, sk_{sig}, ADM, and grp, draw γ nonces: $s_i = r_i \leftarrow \mathcal{R}$, where $\mathcal{R}$ is the randomness space of the chameleon-hash used. and γ additional tags, i.e.,* $\mathsf{TAG}_i \leftarrow \{0,1\}^{2\lambda}$. *Let:*

$$h[i] \leftarrow CHash(pk_{san}.pk_c, \mathsf{TAG}_i, (i, grp[i], \mathsf{TAG}_i, pk_{sig}), r_i)$$

for all $i = 1, \ldots, \gamma$.
Generate:

$$\sigma_c \leftarrow \mathsf{SSign}(sk_s, (h[1], \ldots, h[\gamma], grp[0], pk_{san}, ADM, grp, (r_i)_{0<i\leq\gamma})$$

and

$$\sigma_d \leftarrow \mathsf{SSign}(sk_s, (h[1], \ldots, h[\gamma], s_1, \ldots, s_{\gamma}, m, pk_{san}))\,.$$

Output:

$$\sigma = (\sigma_c, \sigma_d, (\mathsf{TAG}_i)_{0<i\leq\gamma}, (r_i)_{(0<i\leq\gamma)}, (s_i)_{0<i\leq\gamma}, ADM, grp)\,.$$

Verification: *On input of pk_{sig}, pk_{san}, m, $\sigma = (\sigma_c, \sigma_d, (\mathsf{TAG}_i)_{0<i\leq\gamma}, (r_i)_{0<i\leq\gamma}, (s_i)_{0<i\leq\gamma}, ADM)$ compute:*
$h[i] \leftarrow CHash(pk_{san}.pk_c, \mathsf{TAG}_i, (i, grp[i], \mathsf{TAG}_i, pk_{sig}), s_i)$.
Check, if σ_d either verifies under $pk_{san}.pk_s$ or pk_{sig}.
If σ_d verifies under pk_{sig}, also check, if the r_i protected by σ_c and σ_d are equal, i.e., if $r_i = s_i$.
If so, output:

$$\mathsf{SVerify}(pk, (h[i]_{0<i\leq\gamma}, grp[0], pk_{san}, ADM, grp, (s_i)_{0<i\leq\gamma}), \sigma_c)\,.$$

Sanitizing: *On input of pk_{sig}, sk_{san}, m, MOD and $\sigma = (\sigma_c, \sigma_d, (\mathsf{TAG}_i)_{0<i\leq\gamma}, (r_i)_{0<i\leq\gamma}, (s_i)_{0<i\leq\gamma}, ADM)$ check if the received message-signature pair is valid using Verify.*
If not, stop and output $\perp$.
*If it verifies: For each **group** grp$[i] \in$ MOD, draw new tags $\mathsf{TAG}'_i \leftarrow \{0,1\}^{2\lambda}$. If grp$[i] \notin$ MOD, set $\mathsf{TAG}'_i = \mathsf{TAG}_i$ and $s'_i = r_i$.*
Afterward, compute:

$$s'_i \leftarrow CHAdapt(sk_{san}.sk_c, \mathsf{TAG}_i, (i, grp[i], \mathsf{TAG}_i, pk_{sig}), r_i,$$

$$\mathsf{TAG}'_i, (i, grp[i]', \mathsf{TAG}'_i, pk_{sig}))\,.$$

	Signing			Verifying			Sanitizing		
ℓ / λ	100	500	1,000	100	500	1,000	100	500	1,000
512 Bit	16	63	125	15	46	78	157	766	1,641
1,024 Bit	28	112	14,132	20	96	22	1,007	4,948	9,720
2,048 Bit	110	391	750	62	328	657	7,109	35,328	70,997
4,096 Bit	563	1,546	2,798	250	1,235	2,469	54,719	272,672	545,062

Table 14. *blockacc*$\mathcal{SSS}^{INT}$: Performance of Construction 3; ℓ is the number of blocks; λ is the security parameter in Bit; median runtime in ms; as jointly published in *de Meer, Pöhls, Posegga, and Samelin* [132]

Output (m', σ'), where $m' \leftarrow \mathsf{MOD}(m)$ and

$$\sigma' = (\sigma_c, \sigma'_d, (\mathsf{TAG}')_{0<i\leq\gamma}, (r_i)_{0<i\leq\gamma}, (s'_i)_{0<i\leq\gamma}, \mathsf{ADM}, \mathsf{grp}),$$

where $\sigma'_d \leftarrow \mathsf{SSign}(sk_{san}.sk_s, (h[1], \ldots, h[\gamma], s'_1, \ldots, s'_\gamma, m', pk_{san}))$, where each $h[i]$ is calculated as $h[i] \leftarrow \mathsf{CHash}(pk_{san}.pk_c, \mathsf{TAG}_i, (i, \mathsf{grp}[i], \mathsf{TAG}_i, pk_{sig}), s_i)$. Note that $r'_i = r_i$, where $\mathsf{grp}[i] \in \mathsf{MOD}$, is only possible with negligible (as a function of λ) probability, if the TAG is changed, which is now appended to the message. Of course, one has to check that each tag is exactly padded to 2λ.

Proof: *Always return $\bot$.*

blkJudge: *On input of $m = (m[1], \ldots, m[\ell])$, $m[i] \in \{0,1\}^*$, and*

$$\sigma = (\sigma_c, \sigma_d, (\mathsf{TAG}_i)_{0<i\leq\gamma}, (r_i)_{0<i\leq\gamma}, (s_i)_{0<i\leq\gamma}, \mathsf{ADM}, \mathsf{grp}),$$

pk_{sig}, pk_{san}, $\bot$ and an index i, first check, if σ verifies. For group $\mathsf{grp}[i] \in \mathsf{grp}$ let:

$$d_i \leftarrow \begin{cases} \mathtt{San} & \text{if } r_i \neq s_i \\ \mathtt{Sig} & \text{else} \end{cases}$$

Output d_i, or $\bot$ on error resp.

Judge: *On input of m,*

$$\sigma = (\sigma_c, \sigma_d, (\mathsf{TAG}_i)_{0<i\leq\gamma}, (r_i)_{0<i\leq\gamma}, (s_i)_{0<i\leq\gamma}, \mathsf{ADM}),$$

pk_{sig}, pk_{san} and $\bot$, first check if σ verifies. For each $\mathsf{grp}[i] \in \mathsf{grp}$, call $d_i \leftarrow \mathsf{grpJudge}(m, \sigma, pk_{sig}, pk_{san}, \bot, i)$. On error, output $\bot$. If $\exists i : d_i \neq \mathtt{Sig}$, then output $\mathtt{San}$ and $\mathtt{Sig}$ otherwise.

13.7.3. *blockacc*$\mathcal{SSS}$ runtime and complexity

For the joint publication [132] (see Appendix A publication n° 16) the constructions were implemented to provide a rough estimate of the runtime. The tests were performed on a desktop with an *Intel* Q9550 Quad Core @2.83 GHz and 3 GiB of RAM. Only one core was utilised for the generation of the signature using RSA as the signature algorithm. The moduli have been fixed to 512, 1,024, 2,048 and 4,096 Bit. Each algorithm was evaluated with 100, 500 and 1,000 blocks. The amount of admissible blocks was fixed to 50% during singing and always all admissible blocks were sanitized during sanitization. Moreover, to maintain comparability, each group is exactly one block of the message, i.e., $\gamma = |\mathsf{ADM}|$. Proof and Judge are very fast, as they contain only a database lookup and are therefore omitted. The results can be seen in Tab. 14, and Tab. 15.

13.7.4. *blockacc*$\mathcal{SSS}$ security proof

The following proofs of *blockacc*$\mathcal{SSS}$'s provable security can also be found in the related joint publication [132] (see Appendix A publication n° 16).

	Signing			Verifying			Sanitizing			Detecting		
ℓ / λ	100	500	1,000	100	500	1,000	100	500	1,000	100	500	1,000
512	16	78	140	15	47	94	172	797	1,578	16	46	94
1,024	47	172	313	31	141	265	1,047	5,062	10,438	32	125	266
2,048	172	516	969	94	437	875	7,547	36,079	72,735	93	421	859
4,096	922	2,157	4,141	328	1,546	3,546	55,453	271,329	562,683	360	1,546	3,109

Table 15. *blockacc$\mathcal{SSS}^{PUB}$*: Performance of Construction 4; ℓ is the number of blocks; λ is the security parameter in Bit; median runtime in ms; as jointly published in *de Meer, Pöhls, Posegga, and Samelin* [132]

Theorem 15 : *blockacc$\mathcal{SSS}^{INT}$* offering transparency from Construction 3 is Secure.

If the underlying signature scheme $\mathcal{SS}$ is unforgeable, the used chameleon hash-function is collision resistant under random tagging attacks, while $\mathcal{PRF}$ and $\mathcal{PRG}$ are pseudorandom, the scheme blockacc$\mathcal{SSS}$ from Construction 3 is private, immutable, group-level non-interactive publicly (INT) accountable and unforgeable.

Proof 15

Each property is proven on its own.

- *Construction 3 is immutable. Let $\mathcal{A}$ denote an efficient adversary breaking the immutability of the presented scheme. One can then construct an adversary $\mathcal{B}$ using $\mathcal{A}$ as a black box to break the unforgeability of the underlying signature scheme as follows. One simulates $\mathcal{A}$'s environment by simulating the signing oracle; the signature of the underlying signature scheme (σ_c) is generated by $\mathcal{B}$'s own oracle. Eventually, $\mathcal{A}$ will output a forgery attempt, i.e., a tuple $(\mathsf{pk}^*, m^*, \sigma^*)$. This finishes the simulation. One will have to distinguish between three cases: (1) One has $\mathsf{pk}_{san} \neq \mathsf{pk}_{san,i}$ for all queries. As pk_{san} has been signed, the underlying signature scheme has been broken. (2) For some j and $i_j \notin \mathsf{ADM}_j$, $m^*_j[j_i] \neq m_j[j_i]$ yields. As m^* has therefore not been queried, the unforgeability of the underlying signature scheme has been broken as well. (3) For some group $\mathsf{grp}_j[i]$, the message has been replaced by a hash-value or vice versa respectively. As this implies $\mathsf{grp}_j[i] \neq \mathsf{grp}^*[i]$, the signature must have been forged, as grp is signed. If neither case happens, the simulation aborts. The signature forgeries can be extracted in all cases and are then returned by $\mathcal{B}$ as a valid forgery of the underlying signature scheme. Hence, $\mathcal{B}$'s success probability equals the one of $\mathcal{A}$.*
- *Construction 3 is transparent. Transparency follows from the definitions of CHash and $\mathsf{CHAdapt}$, as the distribution of r' and h are computationally indistinguishable from uniform [64]. Moreover, the pseudorandom generators output numbers which are computationally indistinguishable from uniform as well. The above does not consider any tag-collisions, as they only appear with negligible probability. Transparency follows.*
- *Construction 3 is group-level sanitizer-accountable. Please note, in the case where $h[i] \neq h^*[i]$, where h denotes the digest of a group, a direct forgery of the underlying signature scheme is implied. This is also true for $\mathsf{pk}_{san,i} \neq \mathsf{pk}^*$ and $\mathsf{ADM}_i \neq \mathsf{ADM}^*$ and $\mathsf{grp}_i \neq \mathsf{grp}^*$. Also note, that in this case the Proof-oracle can trivially be simulated by picking κ itself. Hence, one can focus on the chameleon hash-function. To be successful, the adversary against group-level signer-accountability needs to make sure that the proof algorithm Proof cannot find at least one non-trivial colliding pair of chameleon hash-value. Hence, one has:*

$$\mathsf{CHash}(\mathsf{pk}_{san}, \mathsf{TAG}_{j,0}, ((\mathsf{TAG}_{j,i})_{0 \leq i \leq \gamma}, m), r_{j,0}) =$$
$$\mathsf{CHash}(\mathsf{pk}^*, \mathsf{TAG}^*_{j,0}, ((\mathsf{TAG}_{j,i})^*_{0 \leq i \leq \gamma}, m^*), r^*_0)$$

for some query j. However, this collision is non-trivial and Proof *can find it, which prohibits the attack. This also applies to the outer chameleon hash-value, protecting against match-and-mix attacks. Building an extractor is straightforward and therefore omitted. Sanitizer-accountability for groups follows.*

- *Construction 3 is group-level signer-accountable. Let $\mathcal{A}$ denote an efficient adversary breaking the group-level signer accountability of the proposed scheme. One then constructs an adversary $\mathcal{B}$ using $\mathcal{A}$ as a black box to break the collision-resistance against random-tagging attacks of the underlying chameleon hash-function in the follow way: As before, $\mathcal{B}$ simulates $\mathcal{A}$'s environment. However, calls to the sanitization oracle are simulated using $\mathcal{B}$'s $OAdapt$-oracle and signed by its own generated signature key pair. Eventually, $\mathcal{A}$ returns $(pk^*, \pi^*, m^*, \sigma^*)$.*

  By definition π^* must contain two (non-trivial) colliding tuples:

$$\begin{aligned}&\mathsf{CHash}(pk_{san}, \mathsf{TAG}_{j,0}, (\mathsf{TAG}_{j,i})_{0 \le i \le \gamma}, r_{j,i}) = \\ &\mathsf{CHash}(pk^*, \mathsf{TAG}_i^*, (\mathsf{TAG}_{j,i})_{0 \le i \le \gamma})^*, r_i^*) .\end{aligned}$$

  This finishes the simulation. Afterwards, $\mathcal{B}$ outputs the colliding tuples. These tuples break the collision-resistance of the chameleon hash-function as the tags are drawn at random. Any tag-collision is therefore only possible with negligible probability. Hence, $\mathcal{B}$'s success probability equals the one of $\mathcal{A}$. Please note that this also applies for the outer chameleon hash-value, protecting against match-and-mix attacks. Building an extractor is straightforward and therefore omitted. Hence, the attack discovered by Gong et al. does not apply here, as the scheme adds an additional chameleon hash-value, protecting the whole message, similar to [216]. Signer*-accountability for groups follows.* □

Theorem 16 : $blockacc\mathcal{SSS}^{PUB}$ offering non-interactive publicly accountability from Construction 4 is Secure.

If the underlying signature scheme $\mathcal{SS}$ is UNF-CMA, while the used chameleon hash-function is collision-resistant under random-tagging attacks, the scheme $\mathsf{blockacc}\mathcal{SSS}$ from Construction 4 is private, immutable, unforgeable and group-level non-interactive publicly accountable.

Following the definitions and [64], it is enough to show that privacy, immutability and group-level non-interactive public accountability hold to prove the security of Construction 4.

Proof 16

The proofs for privacy, immutability and unforgeability are exactly the same as for the transparent Construction 3, with two notable exceptions: Transparency is not achieved, as the construction signs the original $r[i]$. However, the randomness does not leak any information about the original message, as the tags are drawn at random. Moreover, the "outer" signature protects against mix-and-match attacks. In other words, the Sanitizer *is only able to draw a new tag, which changes the random coin, but not the message, while the random coins for the chameleon hash-functions are always distributed uniformly, which implies privacy.*

Therefore, one only needs to show that Construction 4 is group-level non-interactive publicly accountable. Assume that there is an efficient adversary $\mathcal{A}$ against group-level non-interactive public accountability. One can then construct an adversary $\mathcal{B}$ using $\mathcal{A}$ as a black box to break the unforgeability of the underlying signature scheme as follows: $\mathcal{B}$ forwards any queries to its own oracles and returns the answers to $\mathcal{A}$. $\mathcal{B}$ also flips a coin $b \leftarrow \{0,1\}$. Eventually, $\mathcal{A}$ returns a tuple (pk^*, m^*, σ^*). If $b = 1$, $\mathcal{B}$ sets $pk_{san} \leftarrow pk^*$ and $(pk_{sig}, sk_{sig}) \leftarrow \mathsf{KGen}_{sig}$ else, $\mathcal{A}$ sets $pk_{sig} \leftarrow pk^*$ and $(pk_{san}, sk_{san}) \leftarrow \mathsf{KGen}_{san}$.

Consequently, there are two distinguishable cases, i.e., a malicious sanitizer and a malicious signer. The probability that the simulation is done for the correct case is exactly $\frac{1}{2}$. Cases where the random coins are equal are omitted, as this only occurs with negligible probability.
Malicious Signer: *As $r_i' \neq r_i$, the underlying signature scheme must been forged, as σ_d protects all r_i, as $r_i' = r_i$ occurs only with negligible probability.*

Malicious Sanitizer: *One knows that $r_i' = r_i$ only occurs with negligible probability. Therefore, σ_d must be a valid forgery.*

In both cases, an extractor can trivially be built. □

13.7.5. $blockacc\mathcal{SSS}^{INT}$ and $blockacc\mathcal{SSS}^{PUB}$ integrity offer and visualisation

The following summarises the integrity offerings of $blockacc\mathcal{SSS}^{PUB}$ and $blockacc\mathcal{SSS}^{INT}$.

13.7.5.1. $blockacc\mathcal{SSS}^{PUB}$ integrity offer and visualisation

$blockacc\mathcal{SSS}^{PUB}$ is a sanitizable signature scheme for an ordered list, e.g. an array. It offers the legally advised public form of accountability, i.e. non-interactive public accountability (PUB). The Verifier only needs public information in order to provide third party verifiable proofs of accountability. It does not offer an accountability on the level of each individual blocks as did $pubacc\mathcal{SSS}^{Block}$ (see Sec. 13.5.4.2). It does go beyond the message-level non-interactive public accountability that was offered by $pubacc\mathcal{SSS}$ (see Sec. 13.5.4.1). $blockacc\mathcal{SSS}^{PUB}$'s level of detection is in-between, i.e. ≥1CD and below BCD, when it comes to the scope of detectability of subsequent modifications as introduced for extended integrity notion (see Sec. 6.3.1 on page 145). However, the current classification and the visualisation given in this thesis does not capture this difference. Summarised, $blockacc\mathcal{SSS}^{PUB}$ classifies as giving ACA – ≥1CD,< BCD – PUB integrity protection.

The Signer is no longer accountable for the whole message after the first successful sanitization of a block, if one assumes the intuitive construction of message-level accountability as an aggregation of all block's block-level property that was given in Definition 185 (see Sec. 13.1.5 on page 315). As accountability is fully achieved for the scope of the whole message $blockacc\mathcal{SSS}$ achieves all levels on the upper half in the visualisation introduced in Sec. 6.9. This was already depicted for $pubacc\mathcal{SSS}$ in Fig. 87 on page 338 and is restated as Fig. 90 here.

13.7.5.2. $blockacc\mathcal{SSS}^{INT}$ integrity offer and visualisation

$blockacc\mathcal{SSS}^{INT}$ is a sanitizable signature scheme for an ordered list, e.g. an array. It offers the interactive non-public form of accountability, i.e. offers transparency (INT). It does go beyond the message-level non-interactive public accountability that was offered by $pubacc\mathcal{SSS}$ (see Sec. 13.5.4.1). However, it does not fully reach the level of individual blocks. Hence, $blockacc\mathcal{SSS}^{INT}$'s level of detection is in-between, i.e. ≥1CD and below BCD, when it comes to the scope of detectability of subsequent modifications as introduced for extended integrity notion (see Sec. 6.3.1 on page 145). The current classification and the visualisation given in this thesis is not capturing this difference.

The Signer is no longer accountable for the whole message after the first successful sanitization of a block, if one assumes the intuitive construction of message-level accountability as an aggregation of all block's block-level property that was given in Definition 185 (see Sec. 13.1.5 on page 315).

Summarised, $blockacc\mathcal{SSS}^{INT}$ classifies as giving ACA – ≥1CD,< BCD – INT integrity protection. This is depicted in Fig. 91.

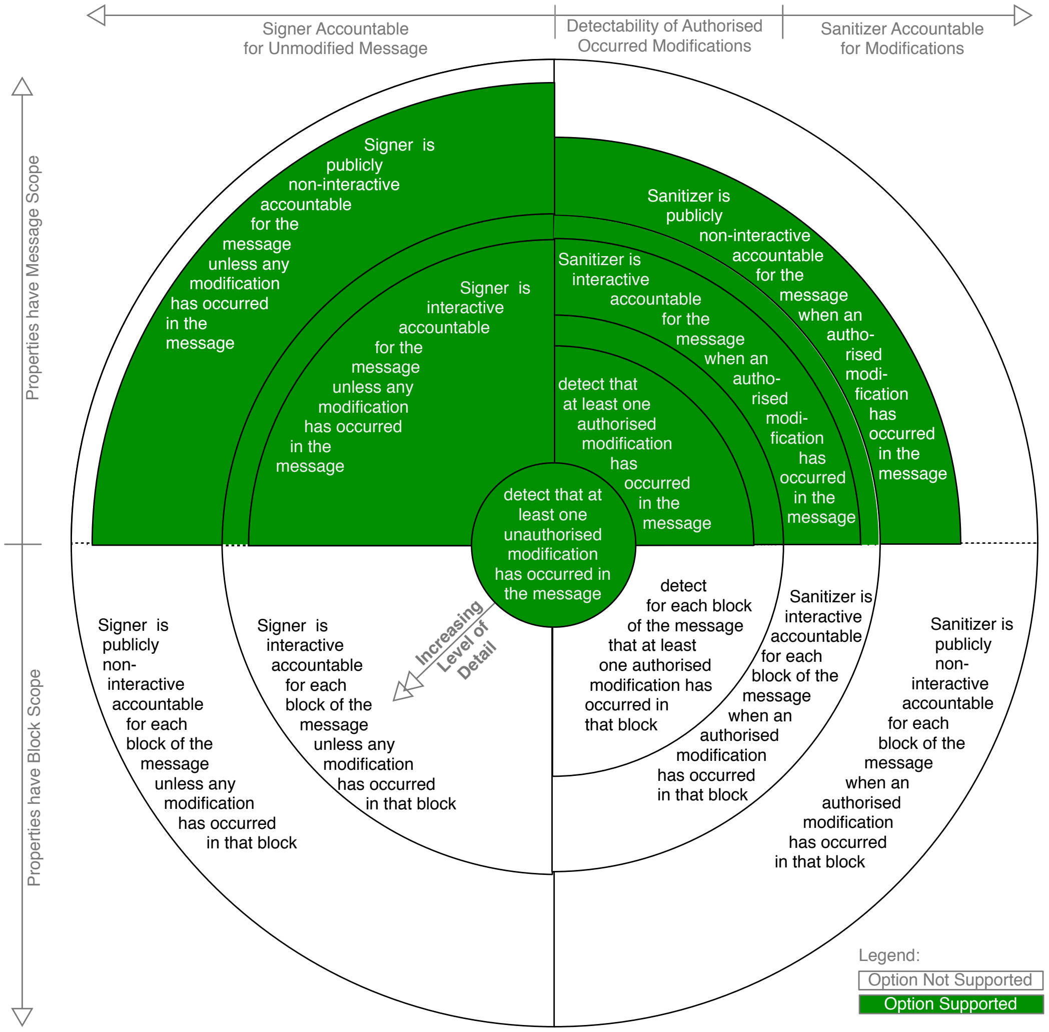

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Increasing Level of Detail
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 90. *blockacc*$\mathcal{SSS}^{PUB}$ offers $\mathrm{ACA}-\geq 1\mathrm{CD}, < \mathrm{BCD}-\mathrm{PUB}$ integrity with giving information about the accountability of groups of blocks, because Verifiers can not always detect it for each single block it does not achieve BCD and thus the lower half of the visualised block-level properties is not achieved

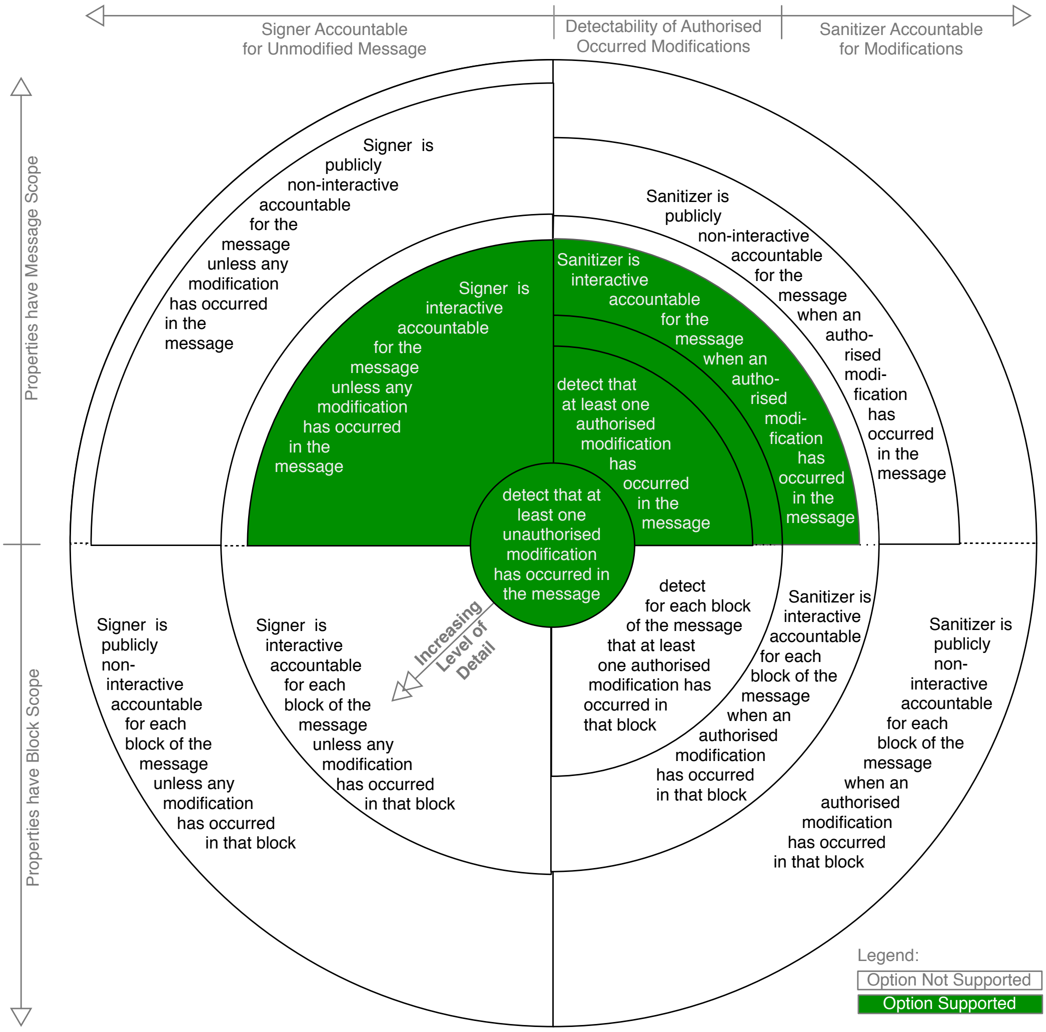

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an authorised modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is interactive accountable for the message when an authorised modification has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Increasing Level of Detail
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 91. *blockacc*$\mathcal{SSS}^{INT}$ offers ACA−≥1CD,<BCD−INT integrity, because Verifiers detect always unauthorized subsequent modifications and accountability is interactive non-public (INT); it does offer more than message-level accountability but not necessarily for each block individually and thus the lower half of the visualised block-level properties is not achieved

13.8. *unlinkable$\mathcal{SSS}$* concept: Strong unlinkability (privacy) and non-interactive public accountability for the message

In the following the scheme *unlinkable$\mathcal{SSS}$* with non-interactive public accountability and perfect unlinkability for an ordered list is presented[802].

13.8.1. *unlinkable$\mathcal{SSS}$* extended security properties

The sanitizable signature scheme *unlinkable$\mathcal{SSS}$* achieves the following security properties (**bold** indicates a new security property introduced in this thesis):

- Unforgeability according to Definition 152, and
- Immutability according to Definition 154, and
- **Perfect Privacy**, and
- Standard Privacy (implied by above Perfect Privacy [69]), and
- **Perfect Unlinkability** according to Definition 194 (see Sec. 13.3.2), and
- optionally **Non-Interactive Public Accountability** according to Definition 180.

Perfect here means that the property holds in an information-theoretical sense. *unlinkable$\mathcal{SSS}$* achieves unlinkability that is robust against malicious or buggy signers. *unlinkable$\mathcal{SSS}$* is more efficient than state-of-the-art schemes, which deploy costly group signatures to achieve unlinkability; *unlinkable$\mathcal{SSS}$* can be constructed using only standard digital signatures, which makes *unlinkable$\mathcal{SSS}$* compatible with existing infrastructure and frameworks.

Finally, *unlinkable$\mathcal{SSS}$* can, simultaneously to the strong notion of perfect unlinkability, achieve the strongest notion of accountability, i.e., non-interactive public accountability and henceforth addresses Shortcoming 1[803] and Shortcoming 3[804].

Of course achieving transparency (INT) is not possible as it is mutually exclusive with non-interactive public accountability (PUB).

13.8.2. *unlinkable$\mathcal{SSS}$* algorithmic description

*unlinkable$\mathcal{SSS}$***'s algorithms are in general like the harmonised SSS algorithms.** The algorithms of *unlinkable$\mathcal{SSS}$* behave in general like the harmonised SSS algorithms presented in Definition 144. *unlinkable$\mathcal{SSS}$* modifies the original definition, which is based on *Brzuska et al.* from PKC '10 [67], to become more robust against malicious or buggy Signers, as well as more robust against a corruption of the Signer's key.

Additionally, the new definition of strengthened unlinkability (Definition 194 given in Sec. 13.3) is more compact than the original one from [67], as the oracles that use the Signer's secret key can now be simulated by the adversary: Sign and Proof oracle from [67] are not required anymore.

Definition 200 : *unlinkable$\mathcal{SSS}$*: Unlinkable and non-interactively publicly accountable SSS

The unlinkable$\mathcal{SSS}$ is an SSS that consists of seven efficient (PPT) algorithms: unlinkable$\mathcal{SSS}$:= $(\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$. All algorithms may output $\perp$ in case of an error.

[802] The results and the construction of *unlinkable$\mathcal{SSS}$* scheme have in part been published as joint work with *C. Brzuska* and *K. Samelin* [69] at EuroPKI 2013 (see Appendix A publication n° 17).

[803] Shortcoming 1: Not all schemes achieve sophisticated levels of privacy (see Sec. 12.1).

[804] Shortcoming 3: No, or no explicit public form of accountability (see Sec. 12.3).

Key Generation: *There are two key generation algorithms, one for the* Signer *and one for the* Sanitizer. *Both create a pair of keys, a private key and the corresponding public key, based on the security parameter* λ*:*

$$(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^\lambda) \quad (\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^\lambda)\,.$$

Signing: *The* Sign *algorithm takes as input the security parameter* λ*, a message* $m = (m[1], \dots, m[\ell])$*,* $m[i] \in \{0,1\}^*$*, the secret key* sk_{sig} *of the signer, the public key* pk_{san} *of the sanitizer, as well as a description* ADM *of the admissibly modifiable blocks. In detail,* ADM *contains a set of indices of the modifiable blocks and the overall number* ℓ *of blocks in* m*, to guard against length-altering attacks. The* Sign *algorithm outputs the message* m *and a signature* σ *(or* $\perp$*, indicating an error):*

$$(m, \sigma) \leftarrow \mathsf{Sign}(1^\lambda, m, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM})\,.$$

Sanitization: *The algorithm* Sanit *takes a message* $m = (m[1], \dots, m[\ell])$*, where* $m[i] \in \{0,1\}^*$*, a signature* σ*, the security parameter* λ*, the public key* pk_{sig} *of the* Signer *and the secret key* sk_{san} *of the* Sanitizer. *It modifies the message* m *according to the modification instruction* MOD*, which contains pairs* $(i, m[i]')$ *describing the index* i *and the block's new value* $m[i]'$*. As a shorthand,* $m' \leftarrow \mathsf{MOD}(m)$ *denotes that the modification instructions* MOD *were successfully applied to create the modified* m' *from* m*. Note,* MOD *could also be just touching the message (see Sec. 3.4.3), i.e.,* $m' = m$ *is generally possible.* Sanit *calculates a new signature* σ' *for the modified message* $m' \leftarrow \mathsf{MOD}(m)$*. The* Sanit *algorithm outputs* m' *and* σ' *(or possibly* $\perp$ *in case of an error):*

$$(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})\,.$$

Verification: *The algorithm* Verify *outputs a decision* $d \in \{\mathtt{false}, \mathtt{true}\}$ *verifying the correctness of a signature* σ *for a message* $m = (m[1], \dots, m[\ell])$*,* $m[i] \in \{0,1\}^*$ *with respect to the public keys* pk_{sig} *and* pk_{san} *and the security parameter* λ*:*

$$d \leftarrow \mathsf{Verify}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san})\,.$$

Proof: *The algorithm* Proof *takes as input the security parameter, the secret signing key* sk_{sig}*, a message* $m = (m[1], \dots, m[\ell])$*,* $m[i] \in \{0,1\}^*$ *and a signature* σ *as well a set of (polynomially bounded) additional message-signature pairs* $\{(m_i, \sigma_i) \mid i \in \mathbb{N}\}$ *and the public key* pk_{san}*. The* Proof *algorithm outputs a string* $\pi \in \{0,1\}^*$ *(or* $\perp$*, indicating an error):*

$$\pi \leftarrow \mathsf{Proof}(1^\lambda, \mathsf{sk}_{sig}, m, \sigma, \{(m_i, \sigma_i) \mid i \in \mathbb{N}\}, \mathsf{pk}_{san})\,.$$

Judge: *The algorithm* Judge *takes as input the security parameter, a message* $m = (m[1], \dots, m[\ell])$*,* $m[i] \in \{0,1\}^*$ *and a valid signature* σ*, the public keys of the parties and a proof* π*. The* Judge *algorithm outputs a decision* $d \in \{\mathtt{Sig}, \mathtt{San}, \perp\}$ *indicating whether the message-signature pair has been created by the signer or the sanitizer (or* $\perp$*, indicating an error):*

$$d \leftarrow \mathsf{Judge}(1^\lambda, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi)\,.$$

13.8.3. *unlinkableSSS* correctness properties

For *unlinkableSSS* the usual correctness properties as described in Sec. 11.4 for SSS are required to hold. In particular, every genuinely signed or sanitized message verifies as valid. Moreover, every genuinely created proof makes the judge decide in favour of the signer. As this is a non-interactively publicly accountable scheme, this must also hold when the proof is empty, i.e. $\pi = \perp$.

13.9. $unlinkable\mathcal{SSS}$ cryptographic instantiation

The results and the construction of $unlinkable\mathcal{SSS}$ scheme have in part been published as joint work with *C. Brzuska* and *K. Samelin* [69] at EuroPKI 2013 (see Appendix A publication n$^{\text{o}}$ 17).

13.9.1. Cryptographic preliminaries

$unlinkable\mathcal{SSS}$ requires a deterministic signature scheme, unforgeable under chosen message attacks (UNF-CMA). Let $\mathsf{S} = (\mathsf{S.KGen}, \mathsf{S.Sign}, \mathsf{S.Verify})$ be such a signature scheme. Deterministic means, that signing identical messages leads to identical signatures, if signed with the same secret key sk. Note, every unforgeable signature scheme can be transformed into a strongly unforgeable and also deterministic scheme using several transformations [34, 209]. An example for a standardised deterministic signature scheme is "RSASSA-PKCS-v1_5-SIGN" [274] or RSASSA-PSS from PKCS-v2.2 [422]. The former scheme was legally recognised if used before the end of 2016, the latter scheme still is recognised (see Sec. 5.3.1).

13.9.2. $unlinkable\mathcal{SSS}$ construction

The general building principle of the inner and outer signature has already been presented for the $pubacc\mathcal{SSS}$ in Sec. 13.5.1 (see especially Fig. 85). For the outer signature $unlinkable\mathcal{SSS}$ uses a standard digital signature. The previous work of *Brzuska et al.* [67] showed that using the anonymity of a group signature scheme makes signatures of the Signer and the Sanitizer indistinguishable, and the non-frameability/traceability property of the group signature scheme assures interactive non-public (INT) accountability. As the inner signature scheme, $unlinkable\mathcal{SSS}$ facilitates a deterministic signature scheme; using this building block it can be proven that the scheme satisfies both, non-interactive public (PUB) accountability and a statistical notion of unlinkability, the strongest notion of privacy. At the same time, the use of a CDSS as a building block maintains a higher compatibility with existing legally recognised schemes than the use of group signatures.[805] It further allows to build upon existing public-key infrastructure (PKI) for the linkage of the Signer's and Sanitizer's public key to the originator[806]. Further, this way of construction means that the resulting scheme only requires a constant number of standard cryptographic operations.

The $unlinkable\mathcal{SSS}$ construction is inspired by the constructions given in [65] and the one for $pubacc\mathcal{SSS}$. It achieves unforgeability, immutability, non-interactive public accountability, perfect privacy, perfect unlinkability and Sanitizer- and Signer-accountability.

Construction 5 : $unlinkable\mathcal{SSS}$

Let $\mathsf{S} = (\mathsf{S.KGen}, \mathsf{S.Sign}, \mathsf{S.Verify})$ be a deterministic and unforgeable signature scheme. Define the sanitizable signature scheme $unlinkable\mathcal{SSS} = (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Judge})$ as follows:

Key Generation: *Algorithm KGen_{sig} generates on input of the security parameter λ a key pair $(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{S.KGen}(1^\lambda)$ of the underlying signature scheme S, and algorithm KGen_{san} for input λ analogously returns a pair $(\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{S.KGen}(1^\lambda)$.*

Signing: *Algorithm Sign on input $m \in \{0,1\}^*$, sk_{sig}, pk_{san}, ADM and computes*

$$\sigma_{\mathsf{fix}} \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{sig}, (0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san})),$$

$$\sigma_{\mathsf{full}} \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{sig}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}))$$

using the underlying signing algorithm. It returns:

$$(m, \sigma) = (m, (\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM}))\,.$$

[805] There is current unpublished joint research together with *F. Höhne* and *K. Samelin* on the legal status of those signatures.

[806] PKI is in general out of scope of this thesis. It is the current legally accepted method to bind a legal entity ti the public key. See discussion on scope in Sec. 1.4.3 and Sec. 7.5.2.

Sanitizing: *Algorithm* Sanit *on input of message* m, *(maybe empty) modification instructions* MOD, *a signature* $\sigma = (\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM})$, *keys* pk_{sig} *and* sk_{san}, *first checks that* MOD *is admissible according to* ADM *and that* σ_{fix} *is a valid signature for message* $(0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san})$ *under key* pk_{sig}. *If not, it stops and outputs* $\perp$. *Else, it generates the modified message* $m' \leftarrow \mathsf{MOD}(m)$ *and computes*

$$\sigma'_{\mathsf{full}} \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{san}, (1, m', \mathsf{pk}_{san}, \mathsf{pk}_{sig}))$$

and outputs $(m', \sigma') = (m', (\sigma_{\mathsf{fix}}, \sigma'_{\mathsf{full}}, \mathsf{ADM}))$.

Verification: *Algorithm* Verify *on input of a message* $m \in \{0,1\}^*$, *a signature* $\sigma = (\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM})$ *and public keys* pk_{sig}, pk_{san} *first checks that* σ_{fix} *is a valid signature for message* $(0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san})$ *under key* pk_{sig} *by checking that* $\mathsf{S.Verify}(\mathsf{pk}_{sig}, (0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san}), \sigma_{\mathsf{fix}}) = \mathtt{true}$. *Second, it returns* `true`, *if:* $\mathsf{S.Verify}(\mathsf{pk}_{sig}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}), \sigma_{\mathsf{full}}) = \mathtt{true}$ *or* $\mathsf{S.Verify}(\mathsf{pk}_{san}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}), \sigma_{\mathsf{full}}) = \mathtt{true}$. *This declares the entire signature as valid. Otherwise it returns* `false`.

Proof: *The* Proof *algorithm always returns* $\perp$.

Judge: Judge *on input of* $m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}$ *and* $\perp$ *parses* σ *as* $(\sigma_{\mathsf{fix}}, \sigma_{\mathsf{full}}, \mathsf{ADM})$ *and it returns* `Sig`, *if:*

$$\mathsf{S.Verify}(\mathsf{pk}_{sig}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}), \sigma_{\mathsf{full}}) = \mathtt{true}.$$

It returns `San`, *if:*

$$\mathsf{S.Verify}(\mathsf{pk}_{san}, (1, m, \mathsf{pk}_{san}, \mathsf{pk}_{sig}), \sigma_{\mathsf{full}}) = \mathtt{true}.$$

If none verifies, it returns $\perp$.

13.9.3. $unlinkable\mathcal{SSS}$ runtime and complexity

For the joint publication [69] at EuroPKI 2013 (see Appendix A publication n° 17), the construction for $unlinkable\mathcal{SSS}$ was implemented.

All tests were performed on an *Intel* T8300 Dual Core @2.40 GHz and 4 GiB of RAM, running *Ubuntu* Version 12.04 LTS (64 Bit) and Java version `1.7.0_03`. For all tests, the messages contained 100, $1,000$ (1k) and $10,000$ (10k) blocks. For each block count, the amount of admissible blocks got fixed to 50%, and always 50% of the admissible blocks were subjected to sanitization, i.e., 25% of all blocks were sanitized. The runtime given is the median of 100 runs. The deterministic signature scheme was “RSASSA-PKCS-v1_5-SIGN” [274][807]. Only a single thread was utilised to calculate the signatures. Obviously, parallelisation could yield performance improvements.

The results of the measurements in Tab. 16 show that $unlinkable\mathcal{SSS}$ keeps a very high performance.

	KeyGen	Sign			Sanit of 25% of ℓ			Verify			blkJudge		
ℓ λ	100/1k/10k	100	1k	10k	100	1k	10k	100	1k	10k	100	1k	10k
2.048 Bit	1,934	22	24	26	13	14	17	1	1	4	1	2	5
4.096 Bit	16,280	149	150	149	78	79	84	4	4	8	5	5	9

Table 16. $unlinkable\mathcal{SSS}$: Performance of Construction 5; ℓ is the number of blocks; median runtime in ms; as jointly published in *Brzuska, Pöhls, and Samelin* [69]

[807] When the runtime measurements where taken, this exact scheme was still legally recognised for the generation of signatures that provide legal value till 2020 (see Sec. 5.3.1 for details). The author assumes that the legally required scheme of RSASSA-PSS from PKCS-v2.2 [422] will not be significantly slower. However, performance will decrease when increasing the key size to comply with the legal recommendations on RSA key sizes. Till 2022, a minimal length of 1976 Bit is prescribed and a length of 2048 bit is recommended [85].

13.9.4. *unlinkable$\mathcal{SSS}$* security proof

The following proofs of *unlinkable$\mathcal{SSS}$*'s provable security can also be found in the related joint publication [69] (see Appendix A publication n^{o} 17).

Theorem 17 : *unlinkable$\mathcal{SSS}$* from Construction 5 is Secure.

> *If the underlying signature scheme S is UNF-CMA and deterministic, then the scheme unlinkable$\mathcal{SSS}$ from Construction 5 is immutable, perfectly private, non-interactive publicly accountable (and therefore* Signer*-/*Sanitizer*-accountable and also unforgeable [64]), i.e., secure.*

The security proofs of Construction 5 (*unlinkable$\mathcal{SSS}$*) follow the ideas of [65] and the ones for *pubacc$\mathcal{SSS}$* (see Sec. 13.5.3 on page 336). From [64, 65] and the work for *pubacc$\mathcal{SSS}$* it yields that non-interactive public accountability, as defined in this thesis (see Definition 180 on page 311), already implies Sanitizer-accountability, Signer-accountability and unforgeability. Moreover, unlinkability implies privacy [65]. Thus, it suffices to prove that Construction 5 is immutable, non-interactive publicly accountable and unlinkable.

Proof 17

Each property is proven on its own.

- *Unlinkability. For two messages m^0 and m^1 with identical fixed parts $m[\mathsf{fix}]$, the signatures σ^0_{fix} and σ^1_{fix} over this part are identical, as Construction 5 uses a deterministic signature scheme. Moreover, the signatures σ^0_{full} and σ^1_{full}, depending on the modifiable message parts, are not used as input for the sanitizing process. Thus, Construction 5 is perfectly unlinkable and perfectly private.*
- *Immutability. Assume towards contradiction that Construction 1 is not immutable. In particular, let $\mathcal{A}$ be an efficient adversary against immutability. An adversary $\mathcal{B}$ against the underlying signature scheme can be constructed as follows: The adversary $\mathcal{B}$ embeds the keys of the signature scheme as the signer's public keys. It then answers $\mathcal{A}$'s queries to the signing oracle by running the algorithm as described in Construction 1, except for signature generation under the signer's key, where $\mathcal{B}$ queries its signing oracle instead of computing them itself. The simulation is perfect. When $\mathcal{A}$ returns $(m^*, \mathsf{pk}^*_{san}, \sigma^*)$, then $\mathcal{B}$ returns $((0, m[\mathsf{fix}], \mathsf{ADM}, \mathsf{pk}_{san}), \sigma_{\mathsf{fix}})$ as a forgery.*

  Further, one needs to prove that $\mathcal{B}$ is successful in attacking the underlying signature scheme, if $\mathcal{A}$ is. Following the definition, $\mathcal{A}$ wins, if it can output a tuple $(m^*, \sigma^*, \mathsf{pk}^*_{san})$ such that $\mathsf{Verify}(m^*, \sigma^*, \mathsf{pk}_{sig}, \mathsf{pk}^*_{san}) = \mathtt{true}$ and $\mathsf{pk}^*_{san} \neq \mathsf{pk}_{san,i}$ for all i queries to the signing oracle or $\exists i, j, j_i \notin \mathsf{ADM} : m^*[j_i] \neq m_i[j_i]$. This implies a forgery against the underlying signature scheme S, as shown next.

  (i) If $\mathsf{pk}^*_{san} \neq \mathsf{pk}_{san\,i}$, then $(0, *, *, \mathsf{pk}^*_{san})$ is fresh.

  (ii) If $\exists i, j, j_i \notin \mathsf{ADM} : m^*[j_i] \neq m_i[j_i]$, then $(0, m[\mathsf{fix}]^*, \mathsf{ADM}, \mathsf{pk}^*_{san})$ is fresh.

  These cases are equal to the attack cases for forgeries of the underlying signature scheme. Thus, $\mathcal{B}$'s success probability is equal to $\mathcal{A}$'s success probability.
- *Non-Interactive Public Accountability. Let $\mathcal{A}$ be an efficient adversary against non-interactive public accountability. Now redefine $\mathcal{B}$ to become another efficient adversary against the unforgeability of the underlying signature scheme S as follows. $\mathcal{B}$ gets as input a public key pk and flips a coin b. If $b = 0$, it sets $\mathsf{pk}_{sig} := \mathsf{pk}$ and runs $\mathsf{S.KGen}$ to generate $(\mathsf{pk}_{san}, \mathsf{sk}_{san})$. If $b = 1$, it sets $\mathsf{pk}_{san} := \mathsf{pk}$ and runs $\mathsf{S.KGen}$ to generate $(\mathsf{pk}_{sig}, \mathsf{sk}_{sig})$. To simulate the oracles for $\mathcal{A}$, the algorithm $\mathcal{B}$ runs the algorithms* Sign *and* Sanit *according to the specification with the exception that whenever a signature is generated under the secret key sk corresponding to pk, $\mathcal{B}$ does not generate the signature itself. Instead, $\mathcal{B}$ queries its signing oracle and passes the result to $\mathcal{A}$. Eventually, the adversary $\mathcal{A}$ outputs a triple $(\mathsf{pk}^*, m^*, \sigma^*)$. One can distinguish between two cases: a malicious* Sanitizer *attack and a malicious* Signer *attack. With probability $\frac{1}{2}$ the simulation was done for the correct case, as in both cases, the output distributions of $\mathcal{B}$'s simulation are identical.*

Malicious Sanitizer
$\mathcal{B}$ *returns* $((0, m[\mathsf{fix}]^*, \mathsf{ADM}, \mathsf{pk}^*_{san}), \sigma_{\mathsf{fix}})$. *As* $m[\mathsf{fix}]^*$ *is fresh, the signing oracle has never signed a message of the form* $(0, m[\mathsf{fix}], \mathsf{ADM}, *)$.

Malicious Signer
$\mathcal{B}$ *returns* $((1, m^*, \mathsf{pk}_{san}, \mathsf{pk}^*_{sig}), \sigma_{\mathsf{full}})$. *As* m^* *is fresh, the signing oracle has never signed a message of the form* $(1, m^*, *, *)$.

Analysis
Thus, the overall success probability of $\mathcal{B}$ *is exactly* $\frac{1}{2}$ *the success probability of* $\mathcal{A}$. □

13.9.5. *unlinkableSSS* integrity offer and visualisation

unlinkableSSS is a sanitizable signature scheme for an ordered list, e.g. an array. It offers the public non-interactive form of accountability (PUB). It does go beyond the standard privacy and additionally provides perfect privacy. Additionally it provides perfect unlinkability which is again beneficial for certain applications and can help to increase privacy. Perfect here means the property holds in an information-theoretical sense. At the same time *unlinkableSSS* provides message-level non-interactive public accountability like it was offered by *pubaccSSS* (see Sec. 13.5.4.1). The Signer is no longer accountable for the whole message after the first successful sanitization of a block and the Sanitizer becomes accountable.

However, the current classification and the visualisation given in this thesis do not capture the added feature of unlinkability. Summarised, *unlinkableSSS* classifies as giving ACA – 1CD – PUB integrity protection. This integrity offer was already visualised in Fig. 87 on page 338 for *pubaccSSS* and is restated in Fig. 92.

13.10. Implementation of an SSS on a secure signature creation device (smart card)

To identify if the SSS can meet Requirement 6[808] this section introduces proof-of-concept implementations of several SSS schemes on a secure smart card[809]. Schemes in this thesis have been designed in order to perform well in practise. Many of them can be constructed with standard cryptographic primitives. Especially, schemes with computational operations that are well supported by standard secure smart cards were preferred due to the analysis given by *Tews and Jacobs* [465][810]. In the following the results are described for the new scheme *pubaccSSS* as jointly published in *Brzuska, Pöhls, and Samelin* [68] (see Appendix A publication n° 11) and presented in Sections 13.4 and 13.5. Further existing schemes got implemented on the same secure smart card: For comparison this section will give the timings, but not re-state the details for the scheme by *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk* [64] and for the scheme by *Brzuska, Fischlin, Lehmann, and Schröder* [65]. Both of them have been implemented on secure smart card and further details can be found in the publication [395] (see Appendix A publication n° 15). The results for all three SSS implementations are presented in Tab. 17.

13.10.1. Secure smart card test environment (same for RSS and SSS)

The test environment consists of the host and the secure smart card. For the implementations this thesis assumed a semi-trusted host. The host is not allowed to have access to secret signature generation or secret sanitization keys. This requires to execute on the secure smart card all those operations which required the use of secrets. This way, the security of the secure smart card allows to strengthen the guarantees that the private information does never leave the secure smart card. And with the secret keys encased inside the secure smart card the physical control over the card allows the signatory to exercise

[808] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

[809] The following section was partly published as joint work with *S. Peters*, *K. Samelin*, *J. Posegga*, and *H. de Meer* at WISTP 2013 [395] (see Appendix A publication n° 15).

[810] *Tews and Jacobs* [465] concluded that big number operation are bad for performance given the capabilities of pure Java Card 2.2.1.

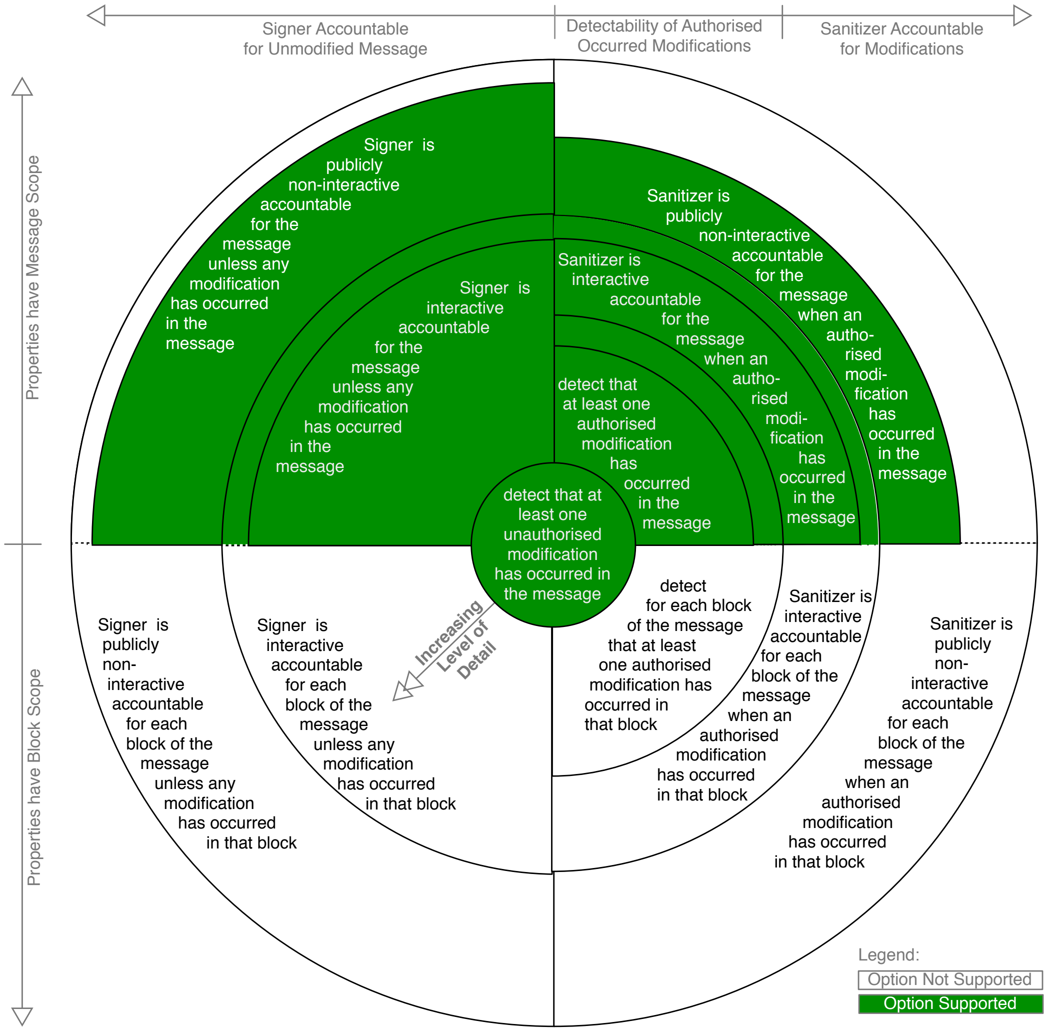

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Increasing Level of Detail
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 92. *unlinkableSSS* offers ACA – 1CD – PUB integrity with added perfect unlinkability, because Verifiers detect always unauthorized subsequent modifications and accountability is public non-interactive (PUB)

control. However the host is still part of the SSCD/QSCD. Especially, it is trusted to correctly compute intermediate values required for a correct signature and correctly transmit these values to the card. For the Signer's host, this includes selection and transmission of the $\mathsf{pk}_{\mathsf{san}}$ to identify the authorized Sanitizer. Like in other signature systems the host is also trusted to correctly display the document to be signed[811]. In the case of MSS not only the document to be signed must be conveyed to the signatory. Additional to the document the host must present the admissible modification policy, ADM, and in the case of SSS, the dedicated Sanitizer to the signatory such that it can be assumed that the signatory knows what it signs.

13.10.1.1. Test parameters

Each party possesses its own secure smart card to execute the algorithm that requires the respective secret keys, i.e., Sign on the Signer's and Sanit on the Sanitizer's secure smart card. Messages get decomposed into blocks of equal length which allows for ADM to be a bitmap, where a block i is designated sanitizable if the value of the bitmap at position i is 1 and otherwise it is immutable.

Each implementation is tested with blocks of 1 byte length and different message lengths of 10, 25 and 50 blocks, each containing the value "`0x11`". Always 50% of the blocks are marked as admissible, such that for blocks $m[1], \ldots, m[\ell]$, $\mathsf{ADM}(i) = 1$, for $i = 0 \pmod 2$. Unless specified otherwise, all admissible blocks are sanitized by replacing the original content of the block with "`0x22`".

[811] See for example [306].

13.10.1.2. Test system

The secure smart card software was written in *Java Card* Ver. 2.2.1 [122]. The secure smart card was a "SmartC@fé® Expert 4.x" from *Giesecke and Devrient* [206]. The host system was an *Intel* i3-2350 Dual Core 2.30 GHz with 4 GiB of RAM. According to the detailed information from *S. Peters* [370], co-author of [395], the secure smart card reader was a SCM SCR335 [436] and it was connected with the PC using USB 1.0.

13.10.1.3. Measurements

All measurements were taken for a complete execution, including those steps performed on the host system. This includes potential additional delays introduced by communication with the secure smart card over the reader. The table omits the time the key generation takes for key pairs with 2048 bit length, as keys are usually generated in advance. All applets were preloaded on the secure smart card before executing any scheme operations on the host and keys were generated prior to executing any algorithm (except algorithms specifically used for key generation of course). Execution times were measured with the method `System.nanoTime`. However, the nanosecond portion of the return value is ignored, effectively only using the microsecond precision of the method. Each algorithm got executed a 100 times in a row and the measured time was divided by the number of executions to obtain the average execution time (median). If the secure smart card was involved in an algorithm's implementation, the connection to the card got opened before each execution and closed it after each run. Therefore the time required for establishing and closing a connection to the card is included in each individual runtime.

13.10.2. $pubacc\mathcal{SSS}^{Block}$ on a smart card

This scheme itself was described in Sections 13.4 and 13.5 and additionally published as joint work [68] (see Appendix A publication n^{o} 11). Its core idea lends itself to an implementation on a secure smart card. To recapitulate: The idea of $pubacc\mathcal{SSS}$'s construction is to split the block to be signed into several groups, one group for immutable blocks and then a group per sanitizable block. The immutable group is signed by the original Signer. Every admissible block is signed separately. Thus additionally, every sanitized block bears the valid signature of the Sanitizer. And every admissible blocks that is still original bears a valid signature of the Signer. This allows for non-interactive public accountability on per block basis. Note, all required operations on the smart card can exploit the card's cryptographic co-processor and should thus become no additional[812] performance obstacle *Tews and Jacobs* [465].

13.10.2.1. Requirements and choice of parameters

The scheme requires a standard digital signature scheme S which is offered by the secure smart card in this case. From the "SmartC@fé® Expert 4.x" secure smart card [206] the `ALG_RSA_SHA_-PKCS1` RSA signature implementation got selected.[813] A key size of 2048 Bit was used to generate the measurements shown in Tab. 11. The message block's hash-values were calculated on the host using the `SHA1withRSA` implementation provided by *SUN*. All the tags chosen by the Signer and Sanitizer got implemented as 1024 Bit numbers chosen by *SUN*'s `SHA1PRNG` implementation of the `java.security.SecureRandom` class.

13.10.3. Secure smart card implementation overview

Fig. 93 taken from the work of *Peters* [370] shows the interaction and flow of commands and answers between the host and a smart card.

[812] Running on a constrained device like a smart card is already a negative performance hit compared to a standard computer or smart phone.

[813] When the runtime measurements where taken this exact scheme was legally still recognised for the generation of signatures that provide legal value till 2020 (see Sec. 5.3.1 for details). The author assumes that the legally required scheme of RSASSA-PSS from PKCS-v2.2 [422] will not be significantly slower. However, keeping up with the legal recommendations on RSA key sizes will decrease performance. Here, the run-time measurements reflect the current legal recommendations, as until the year 2022 a minimal length of 1976 Bit is prescribed and a length of 2048 bit is recommended [85].

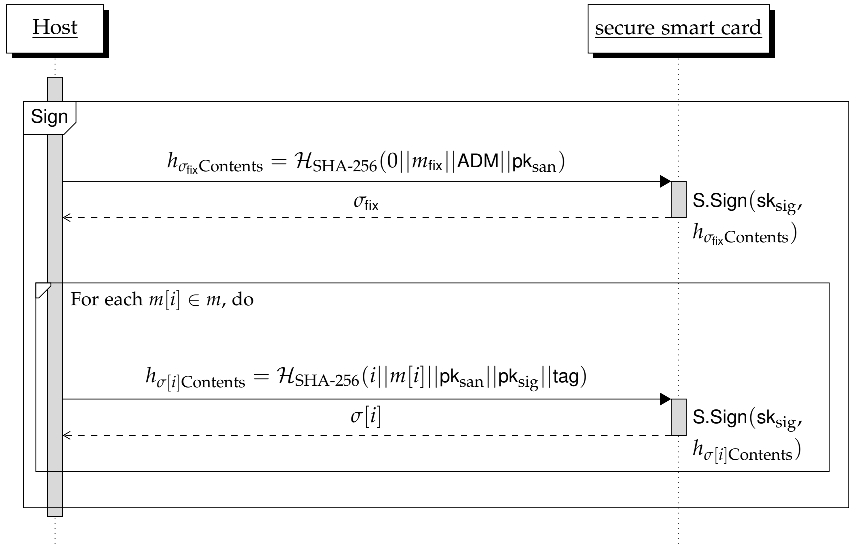

Host
secure smart card
Sign
$h_{\sigma_{\text{fix}}\text{Contents}} = \mathcal{H}_{\text{SHA-256}}(0||m_{\text{fix}}||\mathsf{ADM}||\mathsf{pk}_{\text{san}})$
σ_{fix}
$\mathsf{S.Sign}(\mathsf{sk}_{\text{sig}}, h_{\sigma_{\text{fix}}\text{Contents}})$
For each $m[i] \in m$, do
$h_{\sigma[i]\text{Contents}} = \mathcal{H}_{\text{SHA-256}}(i||m[i]||\mathsf{pk}_{\text{san}}||\mathsf{pk}_{\text{sig}}||\mathsf{tag})$
$\sigma[i]$
$\mathsf{S.Sign}(\mathsf{sk}_{\text{sig}}, h_{\sigma[i]\text{Contents}})$

Figure 93. $pubacc\mathcal{SSS}^{Block}$: Interactions between host and secure smart card during the Sign algorithm; from [370] but notation adjusted

Sign The interactions map onto the actual algorithms for the block-level non-interactively publicly accountable $pubacc\mathcal{SSS}^{Block}$ and correspond to the parameters as described in Sec. 13.5.1.2. To prepare the first communication with the secure smart card for the Sign algorithm the host computes all the relevant hashes over the parameters. When instructed by the host, the secure smart card signs the hash-values using sk_{sig} and returns the signature. First, this is done to obtain σ_{fix}. Then, repeatedly for every admissible blocks, the host generates the SHA-256 hash-value of $(i||m[i]||\mathsf{pk}_{\text{san}}||\mathsf{pk}_{\text{sig}}||\mathsf{tag})$ and requests the secure smart card to return $\sigma[i]$. So the lower half of the interaction depicted in Fig. 93 is executed — potentially — in a loop until all $\sigma[i]$ are returned for all the admissible blocks. The host finally assembles the message's signature containing σ_{fix}, all $\sigma[i])_{i \in \mathsf{ADM}}$ and ADM.

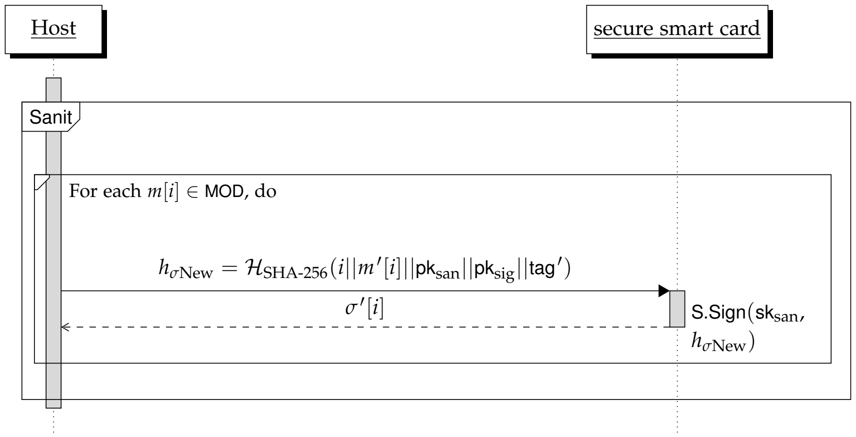

Host
secure smart card
Sanit
For each $m[i] \in \mathsf{MOD}$, do
$h_{\sigma\text{New}} = \mathcal{H}_{\text{SHA-256}}(i||m'[i]||\mathsf{pk}_{\text{san}}||\mathsf{pk}_{\text{sig}}||\mathsf{tag}')$
$\sigma'[i]$
$\mathsf{S.Sign}(\mathsf{sk}_{\text{san}}, h_{\sigma\text{New}})$

Figure 94. $pubacc\mathcal{SSS}^{Block}$: Interactions between host and secure smart card during the Sanit algorithm; from [370] but notation adjusted

Sanit The flow for the Sanit algorithm is depicted in Fig. 94 taken from [370]. The interactions follow directly from the algorithm's description and the required parameter as described in Sec. 13.5.1.2. The host will compute for each to-be-modifiedblock $m[i]$ the new block's information $(i||m'[i]||\mathsf{pk}_{\text{san}}||\mathsf{pk}_{\text{sig}}||\mathsf{tag}')$ and hash it using SHA-256. This is sent to the secure smart card The secure smart card returns the signed answer computed using the secret key sk_{san}. This constitutes the one adapted $\sigma'[i]$ which the host will use to assemble the sanitized signature for the sanitized message which now contains $m'[i]$ instead of $m[i]$.

13.10.4. Performance evaluation of $pubacc\mathcal{SSS}^{Block}$ and other schemes on a secure smart card

	Sign**			Sanit**			Verify		
ℓ	10	25	50	10	25	50	10	25	50
Brzuska et al. [64]	1.22	1.25	1.25	4.25	9.40	17.96	1.09	1.11	1.12
Brzuska et al. [65]	1.09	1.09	1.08	0.58	0.57	0.57	0.017	0.017	0.017
$pubacc\mathcal{SSS}$	3.12	7.16	13.24	2.60	6.65	12.74	0.016	0.039	0.084

	Judge			Detect/Proof		
ℓ	10	25	50	10	25	50
Brzuska et al. [64]	1.78	1.77	1.76	1.53**	1.54**	1.57**
Brzuska et al. [65]	0.017	0.017	0.017	-*	-*	-*
$pubacc\mathcal{SSS}$	0.043	0.051	0.060	0.001	0.001	0.002

*algorithm not defined by scheme **involves secure smart card operations

Table 17. Performance of SSS prototypes on secure smart card ("SmartC@fé® Expert 4.x" [206]); median runtime in seconds; as jointly published in *Pöhls, Peters, Samelin, Posegga, and de Meer* [395]

For all algorithms the tests always assumed the worst case, e.g., the blkJudge algorithm was not randomly choosing which key to use, but the algorithm always first tried to verify the block under the Signer's public key and then under the Sanitizer's public key. When comparing the results listed in Tab. 17 one notices that the non-transparent scheme of *Brzuska et al.* [65] is not affected by the increased amounts of blocks, this is because it works only on the message level. The $pubacc\mathcal{SSS}^{Block}$ has to generate $\lfloor \frac{1}{2}\ell \rfloor + 1$ signatures, as half of the ℓ blocks ($\lfloor \frac{1}{2}\ell \rfloor$) were admissible or sanitized during the performance measurements. With $\ell = 10$ the block-level accountable scheme $pubacc\mathcal{SSS}^{Block}$ actually computes 6 signatures on the secure smart card. The message-level scheme of *Brzuska et al.* [65] only needs 2 signatures regardless of the overall or admissible number of blocks.

14 —— Proposed private RSS with an increased probative value

Overview of Chapter 14

Chapter 14 introduces six newly proposed or refined properties for RSS that go beyond the state of the art. Each property is accompanied by one of the five proposed RSS constructions. Each of these constructions gets proven to achieve the desired new property in this chapter. Each RSS scheme also got published to timely disseminate the results. Tab. 18 presents an overview.

New / refined property	Scheme's name	Section	Publication(s)
Flexibility: Privacy and non-leaf node redaction for tree structured data	$priv\mathcal{RSS}$	14.5	[425]
Flexibility: Signer-controlled structural integrity protection for tree structured data with flexibility to redact intermediate nodes	$flex\mathcal{RSS}$	14.7	[394]
Flexibility: Redact structural information separate from content	$struct\mathcal{RSS}$	14.9	[426]
MRSS: Mergeability of two 'versions' to undo a redaction (14.2) and linkability of 'versions' of redacted documents (14.4)	$merge\mathcal{RSS}$	14.11	[387], [393]
ARSS: Accountability for RSS (14.1)	$pubacc\mathcal{RSS}$	14.13	[388]
Unlinkability of documents created by subsequent redactions (14.3)	$pubacc\mathcal{RSS}$	14.13	[388]

Table 18. Overview of new RSS properties and the five new RSS schemes presented in this thesis

The proposed $priv\mathcal{RSS}$ scheme provides an efficient and private solution for protecting structural and content integrity of tree-structured data. It allows the flexibility of non-leaf node redactions and provides standard privacy. Previous schemes failed in this respect, as attacks on existing RSS for tree-structured data have shown (see Chapter 12).

$flex\mathcal{RSS}$ and $struct\mathcal{RSS}$ add flexibility and control to RSS that protect structured data. $flex\mathcal{RSS}$ adds explicit Signer control for the structural protection offered — or not offered — by the scheme (see Sec. 12.5.1 for shortcomings it overcomes). $struct\mathcal{RSS}$ makes the structure itself a first-class citizen for redaction, e.g., the order in lists gets protected individually and thus becomes a separate privately redactable information (see Sec. 12.5.2 for gains).

$merge\mathcal{RSS}$ allows for merging two different redacted messages, both with valid signatures, only if both messages were created by correctly applying redactions to a common 'ancestor' message. This hides any previous split of documents created by redaction or undoes a redaction through an aggregation with a version that holds the information. Finally, in Sec. 14.15 a tailored instantiation of $merge\mathcal{RSS}$ on a smart card [395] (see Appendix A publication n^{o} 15) demonstrates how to meet Requirement 6[814] by execution inside a QSCD.

$pubacc\mathcal{RSS}$ adds accountability to an RSS such that the once public operation of redaction is no longer public, but depends on a secret distributable to a dedicated Sanitizer. The framework ARSS adds accountability as known from SSS to any RSS.

[814] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

14.1. Proposed RSS property: Accountability and accountable redactable signature scheme (ARSS)

As aforementioned for SSS (see Sec. 11.6.6), accountability allows the Verifier to derive which party is accountable for a given message-signature pair (m, σ). It was not mentioned as a property for RSS, but only in the context of SSS discussed in existing work. This section[815] will thus discuss the introduction of this property for RSS.

This property has been formally introduced for SSSs by *Brzuska et al.* [64]. The definition from there was as follows:

Definition 107 : Accountability

> ***Accountability*** *allows a predefined set of entities to determine a valid signature's origin (Signer or Sanitizer) according to a protocol, i.e., decide which party is accountable (split in Signer-accountability and Sanitizer-accountability) for the signature of a given valid message-signature pair* (m, σ).

In this thesis accountability is differentiated into two forms: interactive non-public (INT) and non-interactive public accountability (PUB) (see Sec. 6.4.3). Accountability is also a legal requirement (see Sec. 7.4). Based on the existing properties an RSS was not offering accountability. Namely, even if the scheme would allow merging and thus offer linkability (see Sec. 14.4 on page 377), the naïve implementation of a Proof/Judge based on linkability would not work as the proof would need leak previously redacted information.[816] Hence, in the existing redactable signature schemes the Signer is the originator for message-signature pairs that verify under the Signer's verification key.

14.1.1. Signer-Accountability for an RSS offering non-interactive public accountability

Assume a redactable signature scheme offers non-interactive public accountability (PUB) but only Signer-accountability. Hence, the Verifier can identify if there was a subsequent modification (authorized as well as unauthorized). If a change is detected, by definition of accountability for SSS on block- and message-level[817], the Signer would be able to repudiate being the originator of a message once it is modified. However, as redaction in standard RSS is public, no accountable party can be identified instead of the Signer. Consequently, nobody is accountable. Contrary, and this may be more intuitive on first sight, one could expect the Signer to be accountable for the remaining message as each of the message's blocks that was not redacted is visible to the Verifier and the redactable signature's unforgeability and immutability properties protected this content against any other subsequent modifications, except the redactions. Correctly, on the level of blocks, the Signer remains the originator of each non-redacted block. Still, a detection of at least one redacted block means that on the message-level a modified message is detected, hence this will make the Signer be able to refute being the originator of that message. This message-level accountability follows only for original, unmodified messages. This lead to the following observation:

If an RSS would offer non-interactive public accountability (PUB) but no Sanitizer-accountability then a Signer stays accountable on the block-level for the remaining, non-redacted, original information even after a redaction has taken place as the remaining blocks' integrity is verifiably unharmed and the Signer is verifiably the block's authentic origin.

[815] The result of added accountability for an RSS has been published, as part of the joint work with *K. Samelin* [388], at ARES 2015 (see Appendix A publication n° 21).

[816] If the secure RSS is transparent but offers linkability between the original and derived redacted messages, the Signer could present a linkable original document as a proof. In particular, the Signer in order to dispute being the originator of an m'' could show a signed document m' where at least one block that is redacted, i.e. not present, in the disputed message m'' is present, i.e. $|m''| < |m'|$ and $\mathsf{Linkable}(1^\lambda, m', \sigma', m'', \sigma'', \mathsf{pk}_{\mathrm{sig}}) = \texttt{true}$. Only due to the linkability of the scheme, this would construct a verifiable proof that a subsequent redaction has happened. However, to prove that a redaction has happened, the Signer needs to hand over an original block in π to Judge that was not known to the Verifier. This breaks standard privacy as the adversary would be given access to a Proof oracle and adversarial queries would reveal information about redacted blocks.

[817] See Sec. 13.2.2 on page 319 and following for the definition of accountability on block-level; and see Sec. 2.10.2.4 on page 35 for the general definition of non-repudiable third-party verifiable Signer-accountability. %

This is in line with the idea that non-repudiation of origin can happen if the alleged originator can show that he did not send this message.

EXAMPLE

For example assume the Signer signed the message "I"|"will"|"not"|"stay"|"at"|"home". Assume also that the RSS scheme used for signing offers 1CD – PUB while it would not allow to see where and how many redactions have taken place. Then the Signer shall not be held accountable for the whole message after it being altered to "I"|"will"|"stay"|"at"|"home". Also this is in line with the relations between block- and message-level properties given in Sec. 13.1.5.

14.1.2. Goal of ARSS: Make RSS Sanitizer- and Signer-accountable and offer non-interactive public accountability (≥1CD – PUB)

The technical Requirement 4[818] stated that a mechanism is required to achieve ACA – ≥ 1CD – INT integrity protection (see Sec. 7.4.1). Without accountability, a secure RSS only yields ACA – UCD integrity protection (see Fig. 110 for $priv\mathcal{RSS}$ and Fig. 113 for $flex\mathcal{RSS}$). This thesis showcases with a construction of $pubacc\mathcal{RSS}$ (see Sec. 14.14) that it is possible to instantiate either a non-interactive publicly accountable RSS (ACA – ≥ 1CD – PUB) or a transparent RSS (ACA – ≥ 1CD – INT). This is achieved by accompanying the RSS with an SSS. Depending on the properties of the facilitated RSS and SSS the resulting combined scheme is transparent (INT) or non-interactively publicly accountable (PUB).

The public form of accountability (ACA – ≥ 1CD – PUB) is helpful to indicate that such an ARSS can behave like legally recognised CDSS (NCA – ≥ 1CD – PUB). This will make it easier to attribute the signed document an increased probative value (see Sec. 17.5). This legal status is of paramount importance to actually deploy a redactable signature scheme in an application domain. Transparency, i.e., the anonymity of the accountable party, contradicts some requirements from the legal side. However, transparency offers additional privacy guarantees which are required if the existence of a redaction must be hidden, e.g., if discrimination may follow from the sheer knowledge that a document was redacted.

Both accountability levels will require the scheme's algorithm to handle additional keys and thus additional parameters in their algorithms. In this thesis these extended RSS schemes are denoted as ARSS. The security definition for accountability for RSS is — even though an ARSS needs slightly extended algorithms — not invasive, i.e. it can be related to prior definitions of RSS properties. In the following the algorithmic framework and the adjusted security model is given. The actual scheme, $pubacc\mathcal{RSS}$, is presented in detail in Sec. 14.13 and Sec. 14.14.

14.1.3. Shortcoming 3 addressed by ARSS

The new property for RSS of *non-interactive public* accountability embodies that the assignment of the party that is accountable can be proven non-interactively and just with the knowledge of public keys and the message-signature pair. This also brings accountability to RSS. Previous, there was not only no explicit public form of accountability for RSS, but there was no notion of Sanitizer-accountability either. To the best of the author's knowledge the property of public accountability was not formalised or discussed for RSS before; and the result presented in this section is part of a publication [388] at ARES 2015, jointly done with *K. Samelin* (see Appendix A publication n^{o} 21).

ARSS directly addresses the Requirement 3[819] on accountability as described in Sec. 7.3. Further, if the scheme — as the later presented $pubacc\mathcal{RSS}$ — can establish accountability publicly and non-interactively, then this also fulfils Requirement 2.3[820]. Thereby it addresses the Shortcoming 3[821] that was described and motivated in Sec. 12.3.[822]. The public accountability allows any third party to im-

[818] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[819] Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification) (see Sec. 7.3).

[820] Requirement 2.3 (Mechanism SHOULD enable Verifier to verify the proof of consent non-interactively and publicly)

[821] Shortcoming 3: No, or no explicit public form of accountability (see Sec. 12.3).

[822] For further motivation see the discussions Sec. 6.4.4 or Sec. 6.4.5.

mediately decide whether a message-signature pair has been issued by the Signer or by the Sanitizer without further interactions or inputs[823]. This public form is especially helpful as it eases to argue that non-interactively publicly accountable MSS offer an increased probative value as their behaviour and functionality becomes close to that provided by standard signatures (see Sec. 12.3 and Sec. 6.4.5).

14.1.4. ARSS algorithmic description

Accountable redactable signatures (ARSS) allow to derive the accountable party of a given message-signature pair (m, σ). Note, to make RSS accountable it becomes a necessity — as in SSS — to introduce a second key pair for the second entity — the accountable Sanitizer of the ARSS. For the same reasons as in SSS, the second key pair has to be independent of the Signer's key pair. As the idea is to achieve accountability through the combination with an SSS, this thesis denotes the second key pair accordingly as $(\mathsf{sk}_{\mathrm{san}}, \mathsf{pk}_{\mathrm{san}})$. The following ARSS definition covers a single redactor/sanitizer only; an extension to multiple ones is beyond the scope of this thesis[824].

Definition 201 : Accountable RSS (ARSS)

An accountable RSS, denoted ARSS, consists of seven efficient (PPT) algorithms. As ARSSs are an extension of RSSs, new algorithms are added, and existing ones altered. In particular, let ARSS $:= (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Redact}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$*, such that:*

Key Generation: *There are two key generation algorithms, one for the Signer and one for the Sanitizer. Both create a pair of keys, a private and the corresponding public key, with respect to the security parameter λ:*

$$(\mathsf{pk}_{sig}, \mathsf{sk}_{sig}) \leftarrow \mathsf{KGen}_{sig}(1^{\lambda}), \ \ (\mathsf{pk}_{san}, \mathsf{sk}_{san}) \leftarrow \mathsf{KGen}_{san}(1^{\lambda}) .$$

Signing: *The Sign algorithm takes as input a message m, sk_{sig}, and pk_{san}. It outputs a signature σ:*

$$\sigma \leftarrow \mathsf{Sign}(1^{\lambda}, m, \mathsf{sk}_{sig}, \mathsf{ADM}, \mathsf{pk}_{san}) .$$

Redacting: *Algorithm Redact takes a message m, MOD, a signature σ, pk_{sig}, and sk_{san}.* mod *is defined as for RSSs. Then, Redact outputs m' and σ':*

$$(m', \sigma') \leftarrow \mathsf{Redact}(1^{\lambda}, m, \mathsf{MOD}, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san}) .$$

Verification: *The Verify algorithm outputs a decision $d \in \{\texttt{false}, \texttt{true}\}$ verifying the correctness of a signature σ for a message m with respect to the public keys pk_{sig} and pk_{san}:*

$$d \leftarrow \mathsf{Verify}(1^{\lambda}, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}) .$$

Proof: *The Proof algorithm takes as input sk_{sig}, a message m and a signature σ as well as a set of (polynomially many) additional message/signature pairs $\{(m_i, \sigma_i)\}$ and pk_{san}. It outputs a string $\pi \in \{0,1\}^*$:*

$$\pi \leftarrow \mathsf{Proof}(1^{\lambda}, \mathsf{sk}_{sig}, m, \sigma, \{(m_i, \sigma_i) \mid i \in \mathbb{N}\}, \mathsf{pk}_{san}) .$$

Judge: *Algorithm Judge takes as input a message m, a signature σ, the public keys of both parties and a proof π. It outputs a decision $d \in \{\texttt{Sig}, \texttt{San}\}$ indicating whether the message/signature pair has been created by the Signer or the Sanitizer:*

$$d \leftarrow \mathsf{Judge}(1^{\lambda}, m, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san}, \pi) .$$

[823] It is always assumed that the Verifier posses the required trusted public verification keys of the Signer and potentially the Sanitizer as stated as Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed) (see Sec. 3.8.3).

[824] The intuition is that multi-redactor capabilities can also be directly inherited from the interwoven SSS and thus can be achieved as multi-sanitizers, e.g., [112].

```
Experiment $\text{Sig-Accountability}_{\mathcal{A}}^{\mathsf{ARSS}}(\lambda)$
    $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
    $b \leftarrow \{0,1\}$
    $(\mathsf{pk}^*, \pi^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Redact}(1^\lambda, \cdots, \mathsf{sk}_{\mathrm{san}})}(1^\lambda, \mathsf{pk}_{\mathrm{san}})$
        let $(m'_i, \sigma'_i)$ for $i = 1, \ldots, q$
        denote the answers from oracle Redact
    return 1, if
        $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}) = \mathtt{true}$ and
        $\forall i \in \{1, 2, \ldots, q\} : (\mathsf{pk}^*, m^*) \neq (\mathsf{pk}_{\mathrm{sig},i}, m'_i)$ and
        $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}, \pi^*) = \mathtt{San}$
```

Figure 95. Signer-Accountability Experiment for ARSS

14.1.5. ARSS correctness properties

For ARSS it is required that all correctness requirements hold (see Sec. 11.11 for details). In a publicly accountable ARSS it is required that Judge outputs the accountable party even on an empty proof ($\pi = \perp$). This is analogous to public accountability for SSS (see Sec. 13.1.2).

14.1.6. ARSS extended security model

Following the general idea of non-interactive public accountability this thesis does not consider transparency to be an integral security requirement. As shown privacy, unforgeability and immutability can be achieved without transparency (see Sec. 11.14). Further, also for RSS — as for SSS — transparency and non-interactive public accountability are mutually exclusive (see Sec. 13.1.3 for SSS).

The properties of unforgeability, immutability and standard privacy of SSS apply to ARSS.

The properties and algorithms offered by an ARSS are identical to that of SSS. To integrate easily, ARSS does not require any other definitions than those given for SSS. They are similar to the ones for SSS given previously[825]:

- unforgeability (Sec. 11.6.1),
- immutability (Sec. 11.6.2), and
- standard privacy (Sec. 11.6.3).

In the definition for the Signer-accountability (see Definition 202), the adversarial Signer has to generate a proof π^* which makes Judge wrongly decide that the Sanitizer is accountable, while in fact the Sanitizer is not. In the definition for the Sanitizer-accountability (see Definition 202, the adversary in the role of the Sanitizer has to generate a message/signature (m^*, σ^*) which makes Proof generate a proof π, leading the Judge to decide that the Signer is accountable, while in fact the Signer is not.

Definition 202 : ARSS Signer-Accountability

> *A redactable signature scheme RSS is* ***Signer-accountable****, if for any efficient adversary $\mathcal{A}$ the probability that the experiment Sig-Accountability$_{\mathcal{A}}^{\mathsf{ARSS}}(\lambda)$ given in Fig. 95 returns 1 is negligible (as a function of λ).*

Definition 203 : ARSS Sanitizer-Accountability

> *A redactable signature scheme RSS is* ***Sanitizer-accountable****, if for any efficient adversary $\mathcal{A}$ the probability that the experiment San-Accountability$_{\mathcal{A}}^{\mathsf{ARSS}}(\lambda)$ given in Fig. 96 returns 1 is negligible (as a function of λ).*

The basic idea for defining public accountability is that an adversary, i.e., the Sanitizer or the Signer, has to be able to make the Judge decide wrongly on an empty proof π. This captures the idea that public accountability allows an outsider to derive the accountable party on its own: it is public as running Proof is not necessary, and Judge does not require any secret material.

[825] They have been restated in the publication [388] for completeness (see Appendix A publication n° 21), but are omitted here for brevity.

```
Experiment $\mathsf{San\text{-}Accountability}_{\mathcal{A}}^{\mathsf{ARSS}}(\lambda)$
    $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
    $b \leftarrow \{0,1\}$
    $(\mathsf{pk}^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}})$
        let $(m_i, \mathsf{pk}_{\mathrm{san},i})$ and $\sigma_i$ for $i = 1, \ldots, q$
        denote the queries and answers to/from oracle Sign
    $\pi \leftarrow \mathsf{Proof}(1^\lambda, \mathsf{sk}_{\mathrm{sig}}, m^*, \sigma^*, \{(m_i, \sigma_i) \mid 0 < i \leq q\}, \mathsf{pk}^*)$
    return 1, if
        $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*) = \mathtt{true}$ and
        $\forall i \in \{1, 2, \ldots, q\} : (\mathsf{pk}^*, m^*) \neq (\mathsf{pk}_{\mathrm{san},i}, m_i)$ and
        $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*, \pi) = \mathtt{Sig}$
```

Figure 96. Sanitizer-Accountability Experiment for ARSS

```
Experiment $\mathsf{PubAccountability}_{\mathcal{A}}^{\mathsf{ARSS}}(\lambda)$
    $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
    $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
    $(\mathsf{pk}^*, m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\mathrm{sig}},\cdots),\mathsf{Redact}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{san}})}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
        Let $(m_i, \mathsf{ADM}_i, \mathsf{pk}_{\mathrm{san},i})$ and $\sigma_i$ for $i = 1, 2, \ldots, q$
        be the queries and answers to and from oracle Sign
        return 1, if
            $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*) = \mathtt{true}$, and
            $\forall i \in \{1, 2, \ldots, q\} : (\mathsf{pk}^*, m^*) \neq (\mathsf{pk}_{\mathrm{san},i}, m_i)$ and
            $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}^*, \perp) = \mathtt{Sig}$
        Let $(m_j, \mathsf{MOD}_j, \sigma_j, \mathsf{pk}_{\mathrm{sig},j})$ and $(m'_j, \sigma'_j)$ for $j = 1, 2, \ldots, q'$
        be the queries and answers to and from oracle Redact
        return 1, if
            $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}) = \mathtt{true}$ and
            $\forall j \in \{1, 2, \ldots, q'\} : (\mathsf{pk}^*, m^*) \neq (\mathsf{pk}_{\mathrm{sig},j}, m'_j)$ and
            $\mathsf{Judge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}^*, \mathsf{pk}_{\mathrm{san}}, \perp) = \mathtt{San}$
```

Figure 97. Public Accountability Experiment for ARSS

Definition 204 : ARSS Public Accountability

A redactable signature scheme ARSS is ***publicly accountable****, if Proof* $= \perp$*, and, if for any efficient adversary* $\mathcal{A}$ *the probability that the experiment* $\mathsf{PubAccountability}_{\mathcal{A}}^{\mathsf{ARSS}}(\lambda)$ *given in Fig. 97 returns* 1 *is negligible (as a function of* λ*).*

14.1.7. Visualisation of ARSS

The impact of the added public form of accountability (PUB) of the Sanitizer is shown in Fig. 98 using the visual representation of the integrity protection.[826] Due to the accountability being non-interactive public (PUB) it goes furthest to the edge of the circle.

[826] In general, the properties protection is depicted by marking the areas grey. From the centre circle towards the outside the visualisation shows an increasing protection scope that includes inner levels. Therefore the outermost level is marked in dark grey while the included levels are marked in light grey.

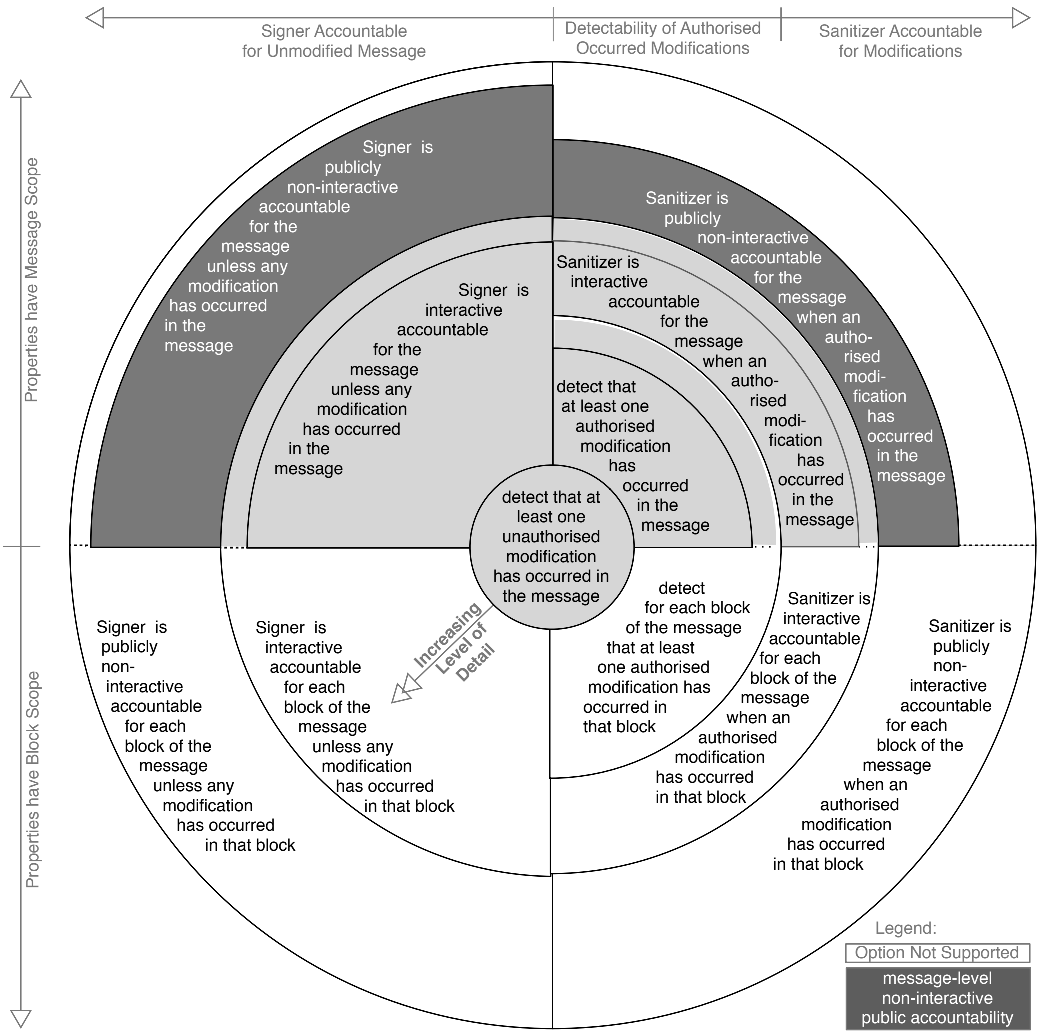

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Increasing Level of Detail
Legend:
Option Not Supported
message-level non-interactive public accountability

Figure 98. Visual representation of an RSS that supports *non-interactive public* accountability of both the Signer and the Sanitizer on *message-level* (ARSS)

14.2. Proposed RSS property: Mergeability, the explicit inverse of redact (MRSS)

The following framework is denoted MRSS and adds new algorithms to an RSS[827]. Especially, it adds the algorithm Merge that formally codifies an explicit inverse operation to redaction. Note, redactions in MRSS still preserve the confidentiality in the same way as an RSS, but works under the assumption that the party already knows a block's original content as it is given two versions' of redacted documents stemming from the same ancestor. Recall Fig. 71 on page 307. This is useful in its own right because it makes it formally explicit if a redactable signature scheme supports this cryptographic operation.

14.2.1. Shortcoming 6 addressed by MRSS

Quite obviously, the MRSS framework is targeted to overcome Shortcoming 6[828]. See Sec. 12.6 on page 306 for a motivation.

[827] This result has been in parts published in the joint work with *H. de Meer*, *J. Posegga* and *K. Samelin* [387] at the Applied Cryptography and Network Security (ACNS) conference in 2014 (see Appendix A publication n° 20).

[828] Shortcoming 6: No explicit dual operation to redact, i.e., no explicit 'merge' of two linkable modified documents.

14.2.2. MRSS algorithmic description

MRSS requires to modify the security model introduced by *Brzuska et al.* [66], as MRSS explicitly allows merging and updating signatures. Thus, MRSS introduces two new algorithms: Merge[829] and Update.

Definition 205 : Mergeable and Updatable RSS (MRSS)

> *A mergeable and updatable RSS, denoted as MRSS, consists of six efficient algorithms. As MRSSs are an extension of RSSs, additional algorithms are added and existing ones are altered. For brevity, without loss of generality, the algorithms for MRSS are in set notation and assume that all block are redactable, i.e. ADM contains always all elements of $\mathcal{S}$. Let MRSS $:= (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Redact}, \mathsf{Update}, \mathsf{Merge})$, such that:*
>
> ***Key Generation:*** *The algorithm KeyGen outputs the public and private key of the signer, i.e., $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, where λ is the security parameter.*
>
> ***Signing:*** *The algorithm Sign gets as input the secret key sk and the set $\mathcal{S}$. It outputs $(\mathcal{S}, \sigma, \tau) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, \mathcal{S})$. Here, τ is a tag.*
>
> ***Redacting:*** *The algorithm Redact takes as input a set $\mathcal{S}$, the public key pk of the Signer, a tag τ, and a valid signature σ and a set $\mathcal{R} \subset \mathcal{S}$ of elements to be redacted. The algorithm outputs $(\mathcal{S}', \sigma', \tau) \leftarrow \mathsf{Redact}(1^\lambda, \mathsf{pk}, \mathcal{S}, \sigma, \mathcal{R}, \tau)$, where $\mathcal{S}' = \mathcal{S} \setminus \mathcal{R}$. $\mathcal{R}$ is allowed to be $\emptyset$. On error, the algorithm outputs $\perp$*
>
> ***Updating:*** *The algorithm Update takes as input a verifying set-signature-tag tuple $(\mathcal{S}, \sigma, \tau)$, the secret key sk and a second set $\mathcal{U}$. It outputs $(\mathcal{S}', \sigma', \tau) \leftarrow \mathsf{Update}(1^\lambda, \mathsf{sk}, \mathcal{S}, \sigma, \mathcal{U}, \tau)$, where $\mathcal{S}' = \mathcal{S} \cup \mathcal{U}$, and σ' is a verifying signature on $\mathcal{S}'$. On error, the algorithm outputs $\perp$.*
>
> ***Merging:*** *The algorithm Merge takes as input the public key pk of the Signer, two sets $\mathcal{S}$ and $\mathcal{V}$, a tag τ, and the corresponding signatures $\sigma_{\mathcal{S}}$ and $\sigma_{\mathcal{V}}$. It is required that $\sigma_{\mathcal{S}}$ and $\sigma_{\mathcal{V}}$ are valid on $\mathcal{S}$ and $\mathcal{V}$. It outputs the merged set-signature-tag tuple $(\mathcal{U}, \sigma_{\mathcal{U}}, \tau) \leftarrow \mathsf{Merge}(1^\lambda, \mathsf{pk}, \mathcal{S}, \sigma_{\mathcal{S}}, \mathcal{V}, \sigma_{\mathcal{V}}, \tau)$, where $\mathcal{U} = \mathcal{S} \cup \mathcal{V}$ and $\sigma_{\mathcal{U}}$ is valid on $\mathcal{U}$. On error, the algorithm outputs $\perp$.*
>
> ***Verification:*** *The algorithm Verify outputs a decision $d \in \{\mathtt{true}, \mathtt{false}\}$ indicating the correctness of the signature σ, w.r.t. pk and τ, protecting $\mathcal{S}$. `true` stands for a valid signature, while `false` indicates an invalid signature. In particular: $d \leftarrow \mathsf{Verify}(1^\lambda, \mathsf{pk}, \mathcal{S}, \sigma, \tau)$.*

This thesis assumes that one can efficiently and uniquely identify all the elements $v_i \in \mathcal{S}$ from a given set $\mathcal{S}$. All algorithms, except Sign and Update, are public operations, as common in RSSs. In other words, all parties can redact and merge sets, which includes the Signer, as well as any intermediate recipient.

Note, τ does not change on any operation. As *merge*$\mathcal{RSS}$ allows merging signatures, unlinkability cannot be achieved: τ makes signatures linkable.

14.2.3. MRSS correctness

The correctness properties for RSS (see Sec. 11.11) must also hold in an MRSS, i.e., every genuinely signed, redacted, merged, or updated set must verify. The same is true for updates and merging signatures. This must even hold transitively, i.e., the history of the signature must not matter.

[829] An algorithm with similar functionality to Merge appears in [314], where it was named "combine" by *Lim et al.*

```
**Experiment** $\mathsf{Unforgeability}_{\mathcal{A}}^{\mathsf{MRSS}}(\lambda)$
  $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  Set $\mathrm{T} \leftarrow \emptyset$
  $(\mathcal{S}^*,\sigma^*,\tau^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot)}_{\mathsf{Update}(1^\lambda,\mathsf{sk},\cdots)}(1^\lambda,\mathsf{pk})$
    For each adaptive query to the oracle Sign:
      let $(\mathcal{S},\sigma,\tau)$ denote the answer from Sign
      Set $\mathcal{S}_\tau \leftarrow \mathcal{S}$
      Set $\mathrm{T} \leftarrow \mathrm{T} \cup \{\tau\}$
    For each call to the oracle Update:
      let $(\mathcal{S},\sigma,\tau)$ denote the answer from Update
      Set $\mathcal{S}_\tau \leftarrow \mathcal{S}_\tau \cup \mathcal{S}$
  return 1, if
    $\mathsf{Verify}(1^\lambda,\mathsf{pk},\mathcal{S}^*,\sigma^*,\tau^*) = \texttt{true}$ and
    $\tau^* \notin \mathrm{T}$ or $\mathcal{S}^* \nsubseteq \mathcal{S}_{\tau^*}$
```

Figure 99. Unforgeability Experiment for MRSS

```
**Experiment** $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{MRSS}}(\lambda)$
  $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{LoRRedact}(1^\lambda,\cdots,\mathsf{sk},b)}_{\mathsf{Update}(1^\lambda,\mathsf{sk},\cdots)}(1^\lambda,\mathsf{pk})$
    where oracle LoRRedact
      for input $\mathcal{S}_0,\mathcal{S}_1,\mathcal{R}_0,\mathcal{R}_1$:
      If $\mathcal{R}_0 \nsubseteq \mathcal{S}_0 \vee \mathcal{R}_1 \nsubseteq \mathcal{S}_1$, return $\perp$
      if $\mathcal{S}_0 \setminus \mathcal{R}_0 \neq \mathcal{S}_1 \setminus \mathcal{R}_1$, return $\perp$
      $(\mathcal{S},\sigma,\tau) \leftarrow \mathsf{Sign}(1^\lambda,\mathsf{sk},\mathcal{S}_b,\tau)$
      return $(\mathcal{S}',\sigma',\tau) \leftarrow \mathsf{Redact}(1^\lambda,\mathsf{pk},\mathcal{S},\sigma,\mathcal{R}_b,\tau)$.
  return 1, if $b = d$
```

Figure 100. Privacy Experiment for MRSS

14.2.4. MRSS extended security model

A mergeable RSS, denoted MRSS adds functionality to an RSS not known before. Hence, the security model (as given in Sec. 11.13) needs to be extended. The following notions are defined next: transparency, privacy, unforgeability, merge privacy, merge transparency, update privacy, and update transparency. Note, τ must not change on any operation. As after merging two sets and their signatures τ makes signatures linkable, unlinkability cannot be achieved in MRSS.

Unforgeability. No one must be able to produce a valid signature on a set $\mathcal{S}^*$ verifying under pk with elements not endorsed by the holder of sk, i.e., the Signer. That is, even if an adversary can adaptively request signatures on different documents, and also can *adaptively update* them, it remains impossible to forge a signature for a new set or new elements not queried. In Fig. 99 $\mathcal{S}_{\tau^*}$ remembers all elements signed by the oracle under tag τ^* and $\mathcal{T}$ collects all tags. This unforgeability definition is analogous to the one for RSS (see also Definition 172 on page 287) and like this in turn analogous to the standard unforgeability requirement of standard digital signature schemes [211] (see also Sec. 2.9.4).

Definition 206 : Unforgeability for MRSS

> *An MRSS is said to be* ***unforgeable****, if for every efficient, i.e., probabilistic polynomial time (PPT), adversary $\mathcal{A}$ the probability that the game depicted in Fig. 99 returns* 1*, is negligible (as a function of λ).*

```
**Experiment** $\mathsf{Transparency}_{\mathcal{A}}^{\mathsf{MRSS}}(\lambda)$
  $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{Sign/Redact}(1^\lambda,\cdots,\mathsf{sk},b),\mathsf{Update}(1^\lambda,\mathsf{sk},\cdots)}(1^\lambda, \mathsf{pk})$
    where oracle Sign/Redact for input $\mathcal{S}, \mathcal{R}$:
      if $\mathcal{R} \nsubseteq \mathcal{S}$, return $\perp$
      $(\mathcal{S}, \sigma, \tau) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, \mathcal{S})$,
      $(\mathcal{S}', \sigma', \tau) \leftarrow \mathsf{Redact}(1^\lambda, \mathsf{pk}, \mathcal{S}, \sigma, \mathcal{R}, \tau)$
      if $b = 1$:
        $(\mathcal{S}', \sigma', \tau) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, \mathcal{S}')$
      return $(\mathcal{S}', \sigma', \tau)$
  return 1, if $b = d$
```

Figure 101. *merge$\mathcal{RSS}$* Transparency Experiment

```
**Experiment** $\mathsf{Merge\ Privacy}_{\mathcal{A}}^{\mathsf{MRSS}}(\lambda)$
  $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{LoRMerge}(1^\lambda,\cdots,\mathsf{sk},b)}_{\mathsf{Update}(1^\lambda,\mathsf{sk},\cdots)}(1^\lambda, \mathsf{pk})$
    where oracle LoRMerge
    for input $\mathcal{S}, \mathcal{R}_0, \mathcal{R}_1$:
      if $\mathcal{R}_0 \nsubseteq \mathcal{S} \vee \mathcal{R}_1 \nsubseteq \mathcal{S}$, return $\perp$
      $(\mathcal{S}, \sigma_{\mathcal{S}}, \tau) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, \mathcal{S})$
      $(\mathcal{S}', \sigma'_{\mathcal{S}}, \tau) \leftarrow \mathsf{Redact}(1^\lambda, \mathsf{pk}, \mathcal{S}, \sigma_{\mathcal{S}}, \mathcal{R}_b, \tau)$
      $(\mathcal{S}'', \sigma''_{\mathcal{S}}, \tau) \leftarrow \mathsf{Redact}(1^\lambda, \mathsf{pk}, \mathcal{S}, \sigma_{\mathcal{S}}, \mathcal{S} \setminus \mathcal{R}_b, \tau)$
      return $\mathsf{Merge}(1^\lambda, \mathsf{pk}, \mathcal{S}', \sigma'_{\mathcal{S}}, \mathcal{S}'', \sigma''_{\mathcal{S}}, \tau)$
  return 1, if $b = d$
```

Figure 102. Merge Privacy Experiment for MRSS

Privacy. The verifier should not be able to gain any knowledge about redacted elements without having access to them. In this definition, the adversary chooses two tuples $(\mathcal{S}_0, \mathcal{R}_0)$ and $(\mathcal{S}_1, \mathcal{R}_1)$, where $\mathcal{R}_i \subseteq \mathcal{S}_i$ describes what shall be removed from $\mathcal{S}_i$. A redaction of $\mathcal{R}_0$ from $\mathcal{S}_0$ is required to result in the same set as redacting $\mathcal{R}_1$ from $\mathcal{S}_1$. The two sets are input to a "Left-or-Right"-oracle which signs $\mathcal{S}_b$ and then redacts $\mathcal{R}_b$. The adversary wins, if it can decide which pair was used by the oracle as the input to create its corresponding output. This is similar to the standard indistinguishability notion for encryption schemes [210].

Definition 207 : Privacy for MRSS

> *An RSS is said to be* ***private****, if for every efficient (PPT) adversary $\mathcal{A}$ the probability that the game depicted in Fig. 100 returns* 1*, is negligibly close to $\frac{1}{2}$ (as a function of λ). Note, this definition does not capture unlinkability.*

Transparency. The verifier should not be able to decide whether a signature has been created by the Signer directly or through the redaction algorithm Redact. The adversary can choose one tuple $(\mathcal{S}, \mathcal{R})$, where $\mathcal{R} \subseteq \mathcal{S}$ describes what shall be removed from $\mathcal{S}$. The pair is input for a "Sign/Redact" oracle that either signs and redacts elements (using Redact) or remove elements as a redaction would do ($\mathcal{S} \setminus \mathcal{R}$) before signing it. The adversary wins, if it can decide which way was taken.

Definition 208 : Transparency for MRSS

> *An RSS is said to be* ***transparent****, if for every efficient (PPT) adversary $\mathcal{A}$, the probability that the game depicted in Fig. 101 returns* 1*, is negligibly close to $\frac{1}{2}$ (as a function of λ).*

```
**Experiment** Merge Transparency$_{\mathcal{A}}^{\mathsf{MRSS}}(\lambda)$
 $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
 $b \xleftarrow{\$} \{0,1\}$
 $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{Sign/Merge}(1^\lambda,\cdots,\mathsf{sk},b)}_{\mathsf{Update}(1^\lambda,\mathsf{sk},\cdots)}(1^\lambda,\mathsf{pk})$
    where oracle Sign/Merge for input $\mathcal{S},\mathcal{R}$:
      if $\mathcal{R} \not\subseteq \mathcal{S}$, return $\perp$
      $(\mathcal{S},\sigma,\tau) \leftarrow \mathsf{Sign}(1^\lambda,\mathsf{sk},\mathcal{S})$
      if $b = 0$:
        $(\mathcal{T}',\sigma'_{\mathcal{T}},\tau) \leftarrow \mathsf{Redact}(1^\lambda,\mathsf{pk},\mathcal{S},\sigma_{\mathcal{S}},\mathcal{R},\tau)$
        $(\mathcal{R}',\sigma'_{\mathcal{R}},\tau) \leftarrow \mathsf{Redact}(1^\lambda,\mathsf{pk},\mathcal{S},\sigma_{\mathcal{S}},\mathcal{S} \setminus \mathcal{R},\tau)$
        $(\mathcal{S}',\sigma',\tau) \leftarrow \mathsf{Merge}(1^\lambda,\mathsf{pk},\mathcal{T}',\sigma'_{\mathcal{T}},\mathcal{R}',\sigma'_{\mathcal{R}},\tau)$
      if $b = 1$: $(\mathcal{S}',\sigma',\tau) \leftarrow (\mathcal{S},\sigma_{\mathcal{S}},\tau)$
      return $(\mathcal{S}',\sigma',\tau)$
 return 1, if $b = d$
```

Figure 103. Merge Transparency Experiment for MRSS

```
**Experiment** Update Privacy$_{\mathcal{A}}^{\mathsf{MRSS}}(\lambda)$
 $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
 $b \xleftarrow{\$} \{0,1\}$
 $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{LoRUpdate}(1^\lambda,\cdots,\mathsf{sk},b)}_{\mathsf{Update}(1^\lambda,\mathsf{sk},\cdots)}(1^\lambda,\mathsf{pk})$
    where oracle LoRUpdate for input $\mathcal{S},\mathcal{R}_0,\mathcal{R}_1$:
      $(\mathcal{S}',\sigma'_{\mathcal{S}},\tau) \leftarrow \mathsf{Sign}(1^\lambda,\mathsf{sk},\mathcal{S} \cup \mathcal{R}_b)$
      return $\mathsf{Update}(1^\lambda,\mathsf{sk},\mathcal{S}',\sigma'_{\mathcal{S}},\mathcal{R}_{1-b},\tau)$
 return 1, if $b = d$
```

Figure 104. Update Privacy Experiment for MRSS

Merge Privacy. If a merged set is given to another third party, the party should not be able to derive any information besides what is contained in the merged set, i.e., a verifier should not be able to decide which elements have been merged from what set. In this definition, the adversary can choose three sets $\mathcal{S},\mathcal{R}_0,\mathcal{R}_1$. The oracle LoRMerge signs $\mathcal{S}$ and then generates two signed redacted versions $\mathcal{S}' = \mathcal{S} \setminus \mathcal{R}_b$ and $\mathcal{S}'' = \mathcal{R}_b$. Then, it merges the signatures again. The adversary wins, if it can decide if $\mathcal{R}_0$ or $\mathcal{R}_1$ was first redacted from $\mathcal{S}$ and then merged back.

Definition 209 : Merge Privacy for MRSS

> *An RSS is **merge private**, if for every efficient (PPT) adversary $\mathcal{A}$, the probability that the game depicted in Fig. 102 returns* 1*, is negligibly close to $\frac{1}{2}$ (as a function of λ).*

Merge Transparency. If a set is given to a third party, the party should not be able to decide whether the set has been created only by Sign or through Sign and Merge. The adversary can choose one tuple $(\mathcal{S},\mathcal{R})$ with $\mathcal{R} \subseteq \mathcal{S}$. This pair is input to a Sign/Merge oracle that signs the set $\mathcal{S}$ and either returns this set-signature pair directly $(b = 1)$ or redacts $\mathcal{S}$ into two signed "halves" $\mathcal{R}$ and $\mathcal{T}$ only to merge them together again and return the set-signature pair derived using Merge $(b = 0)$. The adversary wins, if it can decide which way was taken.

Definition 210 : Merge Transparency for MRSS

> *An RSS is **merge transparent**, if for every efficient (PPT) adversary $\mathcal{A}$, the probability that the game depicted in Fig. 103 returns* 1*, is negligibly close to $\frac{1}{2}$ (as a function of λ).*

Note, on purpose the notions of merge transparency and merge privacy are very similar to the notions of privacy and transparency, as they achieve comparable goals.

Update Privacy. If an updated set is given to another third party, the party should not be able to derive which elements have been added. In the game, the adversary wins, if it can decide which elements were added after signature generation. In this definition, the adversary can choose three sets $\mathcal{S},\mathcal{R}_0,\mathcal{R}_1$. The oracle LoRUpdate signs $\mathcal{S} \cup \mathcal{R}_b$ and then adds $\mathcal{R}_{b-1}$ to the signature. The adversary wins if it can decide which set was used for the update.

```
$\textbf{Experiment}$ $\mathsf{Update\ Transparency}_{\mathcal{A}}^{\mathsf{MRSS}}(\lambda)$
 $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
 $b \xleftarrow{\$} \{0,1\}$
 $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{Sign/Update}(1^\lambda,\cdots,\mathsf{sk},b)}_{\mathsf{Update}(1^\lambda,\mathsf{sk},\cdots)}(1^\lambda, \mathsf{pk})$
    where oracle $\mathsf{Sign/Update}$ for input $\mathcal{S}, \mathcal{R}$:
      if $b = 1$: $(\mathcal{S}', \sigma', \tau) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, \mathcal{S} \cup \mathcal{R})$,
      if $b = 0$: $(\mathcal{T}', \sigma'_{\mathcal{T}}, \tau) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, \mathcal{S})$
              $(\mathcal{S}', \sigma', \tau) \leftarrow \mathsf{Update}(1^\lambda, \mathsf{sk}, \mathcal{T}', \sigma'_{\mathcal{T}}, \mathcal{R}, \tau)$
    return $(\mathcal{S}', \sigma', \tau)$
 return 1, if $b = d$
```

Figure 105. Update Transparency Experiment for MRSS

Definition 211 : Update Privacy for MRSS

> *A scheme RSS is* ***update private****, if for every efficient (PPT) adversary $\mathcal{A}$, the probability that the game depicted in Fig. 104 returns* 1*, is negligibly close to $\frac{1}{2}$ (as a function of λ).*

Update Transparency. A verifying party should not be able to decide whether the received set has been created by Sign or through Update. The adversary can choose one pair $(\mathcal{S}, \mathcal{R})$. This pair is input to a Sign/Update oracle that either signs the set $\mathcal{S} \cup \mathcal{R}$ ($b = 1$) or signs $\mathcal{S}$ and then adds $\mathcal{R}$ using Update ($b = 0$). The adversary wins, if it can decide which way was taken.

Definition 212 : Update Transparency for MRSS

> *A scheme RSS is said to be* ***update transparent****, if for every efficient (PPT) adversary $\mathcal{A}$, the probability that the game depicted in Fig. 105 returns* 1*, is negligibly close to $\frac{1}{2}$.*

As before, the notions of update transparency and update privacy are, on purpose, kept very similar to the notions of privacy and transparency due to their similar goals.

Merge Secure.

Definition 213 : Merge Secure for MRSS

> *An RSS is* ***merge*** *secure, if it is unforgeable, transparent, private, merge transparent, merge private, update private, and update transparent.*

14.2.5. Relation between MRSS properties and other security properties

This section gives the most relevant relations between the security properties. This section can be kept brief, as the new definitions are — by design — similar (in terms of relation) to the ones presented for RSS in Sec. 11.14 based on the ones formalised by *Brzuska et al.* [66]. One must explicitly consider the update-oracle, as it may leak information about the secret key sk.

Theorem 18 : Merge Transparency $\Rightarrow$ Merge Privacy

> *Every scheme which is merge transparent is also merge private.*

Proof 18

> *Intuitively, the proof formalises the following idea: If an adversary can decide which elements have been merged, then it can decide that the signature cannot be created by Sign, but by Merge.*
>
> *Assume an (efficient) adversary $\mathcal{A}$ that wins the game for merge privacy, i.e. breaks it, with probability $\frac{1}{2} + \epsilon$. One can then construct an (efficient) adversary $\mathcal{B}$ which wins the merge transparency game with probability $\frac{1}{2} + \frac{\epsilon}{2}$. According to the merge transparency game, $\mathcal{B}$ receives a public key pk and oracle access to $\mathcal{O}^{\mathsf{Sign}}$, $\mathcal{O}^{\mathsf{Sign/Merge}}$, and $\mathcal{O}^{\mathsf{Update}}$. Let $\mathcal{B}$ randomly pick a bit $b' \in \{0,1\}$. $\mathcal{B}$ forwards pk to $\mathcal{A}$. Whenever $\mathcal{A}$ requests access to the signing oracle $\mathcal{O}^{\mathsf{Sign}}$, $\mathcal{B}$ honestly forwards the query to its oracle and returns the unmodified answer to $\mathcal{A}$. The same is true for $\mathcal{O}^{\mathsf{Update}}$. When $\mathcal{A}$ requests access to $\mathcal{O}^{\mathsf{LoRMerge}}$, i.e.,*

when it sends a query $(\mathcal{S}, \mathcal{R}_0, \mathcal{R}_1)$, then $\mathcal{B}$ checks that $\mathcal{R}_0 \subset \mathcal{S} \wedge \mathcal{R}_1 \subset \mathcal{S}$ and forwards $(\mathcal{S}, \mathcal{R}_{b'})$ to $\mathcal{O}^{\mathsf{Sign/Merge}}$ and returns the answer to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs its guess d. The adversary $\mathcal{B}$ outputs 0, if $d = b'$ and 1 otherwise. What is the probability that $\mathcal{B}$ is correct? Two cases need to be considered:

1. *If $b = 0$, then $\mathcal{O}^{\mathsf{Sign/Merge}}$ signs, redacts, and merges the set. This gives exactly the same answer as $\mathcal{O}^{\mathsf{LoRRedact}}$ would do, if using the bit b'. Hence, $\mathcal{A}$ can correctly guess the bit b' with probability at least $\frac{1}{2} + \epsilon$, if $b = 0$.*
2. *If $b = 1$, then $\mathcal{O}^{\mathsf{Sign/Merge}}$ always signs the set as is. Hence, the answer is independent of b'. $\Pr[\mathcal{B} = 1 \mid b = 1] = \frac{1}{2}$ follows.*

Hence, due to the probability of $\frac{1}{2}$ that $b = 1$, it follows that $\Pr[\mathcal{B} = b] = \frac{1}{2} + \frac{\epsilon}{2}$. As a consequence, $\mathcal{B}$ has non-negligible advantage, if ϵ is non-negligible. □

Theorem 19 : Merge Privacy $\not\Rightarrow$ Merge Transparency

There is a scheme which is merge private, but not merge transparent.

Proof 19

One modifies a secure scheme as follows: At sign, one appends a bit $d = 0$. For all other algorithms d is saved and cut off before the unchanged algorithm is run, and appended after the algorithm finished. However, one sets $d = 1$ after signatures are merged. Obviously, this leaves all other properties, but merge transparency, intact. □

Theorem 20 : Update Transparency $\Rightarrow$ Update Privacy

Every scheme which is update transparent, is also update private.

Proof 20

The proof is essentially the same as for Th. 18.

Theorem 21 : Update Privacy $\not\Rightarrow$ Update Transparency

There is a scheme which is update private, but not update transparent.

Proof 21

The proof is essentially the same as for Th. 19.

Theorem 22 : Merge Transparency is independent

There is a scheme which fulfils all mentioned security goals but merge transparency.

Proof 22

The proof is essentially the same as for Th. 19.

Theorem 23 : Update Transparency is independent

There is a scheme which fulfils all mentioned security goals but update transparency.

Proof 23

The proof is essentially the same as for Th. 19.

Theorem 24 : Unforgeability is independent.

There is a scheme which fulfils all mentioned security goals but unforgeability.

Proof 24

One modifies a secure scheme such that it uses a verify algorithm which always accepts all inputs. □

Theorem 25 : Transparency $\Rightarrow$ Privacy

Every scheme which is transparent, is also private.

Proof 25

Similar to Brzuska et al. [66], thus not re-stated.

Theorem 26 : Privacy $\not\Rightarrow$ Transparency

There is a scheme which is private, but not transparent.

Proof 26

Similar to Brzuska et al. [66], thus not re-stated.

Theorem 27 : Transparency is independent.

There is a scheme which fulfils all mentioned security goals but transparency.

Proof 27

Similar to Brzuska et al. [66], thus not re-stated.

14.3. Proposed RSS property: Unlinkability

This result has been published, as part of the joint work with *K. Samelin* [388], at ARES 2015 (see Appendix A publication n$^{\mathrm{o}}$ 21).

Another strengthened privacy property is denoted unlinkability. Initially this property has been described for SSS by *Brzuska, Fischlin, Freudenreich, Lehmann, Page, Schelbert, Schröder, and Volk* [64]. In the work on RSS by *Brzuska, Busch, Dagdelen, Fischlin, Franz, Katzenbeisser, Manulis, Onete, Peter, Poettering, and Schröder* [66] this property is not described. Closest to this work in respect to this property for RSS is the work by *Camenisch, Dubovitskaya, Haralambiev, and Kohlweiss* [105] related to credential systems and universal composability (UC) from 2015; it was independently published in the same year as [388].

Definition 214 : Unlinkability

***Unlinkability** makes it impossible for third parties to decide from two message-signature pairs if they come from one single source message-signature pair.*

Note: Information leakage through the message itself is out of scope.
Note: Unlinkability is stronger than privacy.

Definition 215 : Unlinkability for RSS

An RSS is unlinkable, if for any efficient adversary $\mathcal{A}$ the probability that the experiment given in Fig. 106 returns 1 *is negligibly close to $\frac{1}{2}$ (as a function of λ).*

Note 1 Information leakage through the message itself is out of scope.
Note 2 Unlinkability is stronger than privacy.

```
Experiment Unlinkability$^{\mathsf{RSS}}_{\mathcal{A}}(\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{LoRRedact}(1^\lambda,\cdots)}(1^\lambda)$
    where oracle LoRRedact on input of
    $m_0, \mathsf{MOD}_0, \sigma_0, m_1, \mathsf{MOD}_1, \sigma_1, \mathsf{pk}_{\mathrm{sig}}$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{Verify}(1^\lambda, m_0, \sigma_0, \mathsf{pk}_{\mathrm{sig}}) \neq \mathsf{Verify}(1^\lambda, m_1, \sigma_1, \mathsf{pk}_{\mathrm{sig}})$, return $\perp$
      return $(m', \sigma') \leftarrow \mathsf{Redact}(1^\lambda, m_b, \mathsf{MOD}_b, \sigma_b, \mathsf{pk}_{\mathrm{sig}})$
  return 1, if $a = b$
```

Figure 106. Unlinkability Experiment for RSS based on *Brzuska et al.* [66]

In the game Unlinkability$^{\mathsf{RSS}}_{\mathcal{A}}(\lambda)$ given in Fig. 106 the adversary can input two message-signature pairs to the oracle. The LoRRedact-oracle then chooses one of the two pairs to generate a new one. To break unlinkability, the adversary has to guess which of the two inputted message-signature pairs was chosen to be sanitized. Note, when compared to the privacy game Privacy$^{\mathsf{RSS}}_{\mathcal{A}}$ given in Definition 174 on page 289 the most noticeable difference is that the signatures are also input to the oracle. They are not internally generated. The adversary has full control of signature generation, hence the adversary also specifies each $\mathsf{pk}_{\mathsf{sig}}$; this follows in line with the formalisation of the strengthened unlinkability for SSS as presented in Sec. 13.3. Unlinkability is independent of transparency (see Construction 11 ($pubacc\mathcal{RSS}$) which is an unlinkable but non-interactive public accountable accountable RSS in Sec. 14.14.1) [388][830]. From the independence of unlinkability from transparency follows that an unlinkable scheme can offer non-interactive public accountability (PUB), which makes it not offering transparency. Both properties, transparency and unlinkability can be seen as stronger privacy properties: For SSS this thesis has shown that unlinkability implies standard privacy (see Sec. 13.8 and in Sec. 13.9)[831]. Thus, it is of interest to also show for RSS that the legally-preferred non-interactive public accountability (PUB) can be offered simultaneously to the strong privacy offered by unlinkability.

14.4. Proposed RSS property: Linkability through explicit mergeability of 'variants' of redacted documents

As an additional algorithm one could want an explicit algorithm that directly indicates if two signed messages are linkable. Under the MRSS framework presented in Sec. 14.2 a mergeable redactable signature scheme that works correctly will allow to use the Merge algorithm to construct a test if two message-signature pair have the same ancestor. This can be used to detect if a message m_1 is a variant of the message m_0 that was created through a valid redaction from a common ancestor of m_0 and m_1.

The description that follows below has been published in the joint technical report [393] and is given in the set notation (see Appendix A publication n° 25). Naively, an algorithm to check linkability can be constructed by first trying to merge two signed sets which are valid under the same public key, and then check if the merged document's signature is valid too. This works because two set-signature-tag tuples are only mergeable if and only if they share a common tag. Which, if from the same Signer, is assumed to be different for every document.

Link: The algorithm Link takes as input a set-signature pair $(\mathcal{S}, \sigma_{\mathcal{S}})$, a public key pk and a second set-signature pair $(\mathcal{T}, \sigma_{\mathcal{T}})$. It outputs a bit $d \in \{0,1\}$, indicating, if $(\mathcal{S}, \sigma_{\mathcal{S}})$ and $(\mathcal{T}, \sigma_{\mathcal{T}})$ have both been derived from the same valid set-signature pair $(\mathcal{U}, \sigma_{\mathcal{U}})$ by Redact or Merge. It outputs 1, if $(\mathcal{S}, \sigma_{\mathcal{S}}) \in \mathsf{span}_{\vdash}(\mathcal{U}, \sigma_{\mathcal{U}}) \wedge (\mathcal{T}, \sigma_{\mathcal{T}}) \in \mathsf{span}_{\vdash}(\mathcal{U}, \sigma_{\mathcal{U}})$ and 0 otherwise.

Note, for an implementation of Link neither $\mathcal{U}$, $\sigma_{\mathcal{U}}$ nor $\mathsf{span}_{\vdash}(\mathcal{U}, \sigma_{\mathcal{U}})$ are required to be generated, which might be costly, this is just for notational purposes.

14.5. $priv\mathcal{RSS}$ concept: Transparent and standard privacy for tree-based documents with intermediary node redaction

This section describes $priv\mathcal{RSS}$[832] which offers a secure and still efficient, i.e. that are in $\mathcal{O}(n)$ with n being the number of nodes in the tree, scheme for an RSS for tree-based documents. Previous state-of-the-art efficient schemes for tree structured data, like the scheme by *Kundu and Bertino*, have been found insecure[833]. $priv\mathcal{RSS}$'s concept builds on ideas of *Kundu and Bertino* [295, 296] to derive a new construction which is provably secure[834] as it does not inherit their flaws. Note, the existing schemes which are secure against those attacks also come with a runtime overhead of $\mathcal{O}(n^2)$. The construction of $priv\mathcal{RSS}$ will set out to keep the better performance while upholding the standard privacy following

[830] This result has been published, as part of the joint work with *K. Samelin*, at ARES 2015 [388] (see Appendix A publication n° 21).

[831] The scheme $unlinkable\mathcal{SSS}$ and the result that unlinkability implies standard privacy and that unlinkability and non-interactive public accountability are not mutually exclusive was jointly published in [69] (see Appendix A publication n° 17).

[832] The result has been published, as part of the joint work with *K. Samelin* and *A. Bilzhause* and *J. Posegga* and *H. de Meer* [425], at ACNS 2012 (see Appendix A publication n° 10).

[833] In Subsections 12.1.4 and 12.2.1 the existing and new attacks are given, together they show that the scheme(s) presented by *Kundu et al.* [295, 296] are insecure with respect to all RSS security properties, i.e. no full integrity protection and no standard privacy.

[834] As defined in Sec. 2.3.1.

Brzuska et al. [66]. Construction 6 (see Sec. 14.6) is secure against the already existing attack that have been shown against such schemes, like the attack by *Brzuska et al.* [66] on the scheme of *Kundu and Bertino* (see Sec. 12.1.4.1). Also $priv\mathcal{RSS}$ is secure against further attacks presented in this thesis in Subsections 12.1.4 and 12.2.1 and published as joint work [425, 426].

14.5.1. $priv\mathcal{RSS}$ extended security properties

Additionally to being private and efficient, $priv\mathcal{RSS}$ allows the Signer to decide, if it is allowed to redact parent or intermediate nodes. The secure redactable signature scheme $priv\mathcal{RSS}$ achieves the following security properties for tree structured data:

- Unforgeability according to Definition 172, and
- Standard privacy following *Brzuska et al.* according to Definition 174, and
- Transparency according to Definition 175.

Additionally it offers the increased functionality of

- Redaction of non-leaf nodes, and
- Consecutive redaction control according to Sec. 14.5.1.1.

The harmonised security properties given in Sec. 11.9 for RSS are based on *Brzuska et al.* [66]. The state-of-the-art model restrictively allows only the cutting of leaves of the tree. The extended functionality of Signer-controlled non-leaf redactability requires an adjustment.

1. *Unforgeability:* No adjustments are required.
2. *Privacy:* The game has been extended to cater for the treatment of tree-based data structures and non-leaf node redactions.

Definition 216 : RSS_T Standard Privacy

> *A redactable signature scheme for trees $\mathcal{RSS}_T$ is* ***private****, if for any efficient (PPT) adversary $\mathcal{A}$, the probability that the game* $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{RSS}_T}(\lambda)$ *shown in Fig. 107 returns* 1*, is negligibly close to* $\frac{1}{2}$ *(as a function of λ).*

```
Experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{RSS}_T}(\lambda)$
  $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $b \xleftarrow{\$} \{0,1\}$
  $d \leftarrow \mathcal{A}^{\mathsf{TSign}(1^\lambda,\mathsf{sk},\cdots),\mathsf{LoRAdapt}(1^\lambda,\cdots,\mathsf{sk},b)}(1^\lambda,\mathsf{pk})$
    where $\mathsf{LoRAdapt}(T_{j,0},\mathcal{N}_{j,0},T_{j,1},\mathcal{N}_{j,1},f_{root},f_{int},\mathsf{sk},b)$
      if $\mathsf{MOD}(T_{j,0},\mathcal{N}_{j,0}) \neq \mathsf{MOD}(T_{j,1},\mathcal{N}_{j,1})$, return $\perp$
      $(T_j,\sigma_{T_j}) \leftarrow \mathsf{TSign}(1^\lambda,\mathsf{sk},T_{j,b},f_{root},f_{int})$
      $(T'_j,\sigma'_{T_j}) \leftarrow \mathsf{TShare}(1^\lambda,T_{j,b},\mathsf{pk},\sigma_{T_{j,b}},\mathcal{N}_{j,b})$
      return $(T'_{j,b},\sigma'_{T,b})$
  return 1, if $b = d$
```

Figure 107. $priv\mathcal{RSS}$ Privacy Experiment

3. *Transparency:* The game has been extended to cater for the treatment of tree-based data structures and non-leaf node redactions.

Definition 217 : RSS_T Transparency

> *A redactable signature scheme for trees $\mathcal{RSS}_T$ is* ***transparent****, if for any efficient (PPT) adversary $\mathcal{A}$, the probability that the game* $\mathsf{Transparency}_{\mathcal{A}}^{\mathsf{RSS}_T}(\lambda)$ *shown in Fig. 108 returns* 1*, is negligibly close to* $\frac{1}{2}$ *(as a function of λ).*

```
$\textbf{Experiment}\ \mathsf{Transparency}_{\mathcal{A}}^{\mathsf{RSS}_T}(\lambda)$
    $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
    $b \xleftarrow{\$} \{0,1\}$
    $d \leftarrow \mathcal{A}^{\mathsf{TSign}(1^\lambda,\mathsf{sk},\cdots),\mathsf{Adapt/Sign}(1^\lambda,\cdots,\mathsf{sk},b)}(1^\lambda,\mathsf{pk})$
        where oracle $\mathsf{Adapt/Sign}$ for input $T,\mathcal{N},f_{root},f_{int}$:
            if $\mathcal{N} \not\subseteq T$, return $\perp$
            $\sigma_T \leftarrow \mathsf{TSign}(1^\lambda,\mathsf{sk},T,f_{root},i)$
            $(T',\sigma_T') \leftarrow \mathsf{TShare}(1^\lambda,T,\mathsf{pk},\sigma_T,\mathcal{N})$
            if $b = 1$: $\sigma_T' \leftarrow \mathsf{TSign}(1^\lambda,\mathsf{sk},T',f_{root},f_{int})$
        return $(T',\sigma_T')$.
    return 1, if $b = d$
```

Figure 108. *privRSS* Transparency Experiment

14.5.1.1. Consecutive redaction / sanitization control: Definition

privRSS offers a property *Haber, Hatano, Honda, Horne, Miyazaki, Sander, Tezoku, and Yao* [222] described based on existing work of *Miyazaki et al.* [347] or *Izu et al.* [264] at the time as follows:

> "It is further desirable to have a scheme that can ensure that certain parts of the document cannot be further redacted, pseudonymized or deidentified. This is important because it serves as a safeguard to prevent certain essential meanings in a document from being changed. This additional requirement for the redaction problem is called "disclosure control," and was introduced by Miyazaki et al. [...]" [222]

Following the idea to harmonise the terminology, this thesis terms this consecutive redaction control. This add the functionality of being able to control the capability to further redact not yet redacted but redactable blocks. Hence the term consecutive. This is equal in functionality to "disclosure condition control" [347] described by *Miyazaki et al.* [347]. It allows to control the redaction operation, which is public, and thus can prohibit an adversary on a correctly redacted document to maliciously additionally "delete[~~s~~] portions he/she deems undesirable, and forward[~~s~~] the additionally sanitized document to the requester." [347]. As the Verifier is not able to detect that several redactions have been carried out, but only sees the result of the sum of all redactions, this lack of control was considered a problem [222, 264, 347].

The property described above in the context of redactions (i.e. removal of data), can be generalised to sanitizations (i.e. when the sanitizer is authorized to do arbitrary modifications[835]). Also *Ma et al.* [325] introduced this for redactions[836]. The following proposed definition is general, not leaning towards RSS. However, the property was designed only into some of the proposed RSS constructions[837].

Definition 218 : Consecutive Redaction / Sanitization Control

An MSS offers ***consecutive redaction / sanitization control****, if the Sanitizer can get (or is by default configuration of the mechanism) authorized by the Signer to make the redaction or sanitization policy stricter. Stricter means that after adjusting ADM to ADM′ the count of all admissible modifications decreases, i.e.*

$$|\mathit{span}_{\vdash}(m,\mathit{ADM}')| < |\mathit{span}_{\vdash}(m,\mathit{ADM})|\,.$$

Moreover, without having access to the previous message-signature pair containing ADM any party seeing (m,σ,ADM') is not able to carry out redactions or sanitizations that have been removed (i.e. contained in $\mathit{ADM} \setminus \mathit{ADM}'$), even if initially authorized.

[835] Following the meaning of the term used in this thesis in accordance with the work of [12] (see Sec. 3.4.3).

[836] Their work allows to initially mark a block as ""Redactable" (RED for short) and "Prohibited to be removed" (PRE for short)" [325]. Then subsequently each redacting Sanitizer could either remove a block when it is marked as redactable (denoted as " "Removed" (REM for short)" [325]) or decide to prohibit its consecutive removal by a subsequent Sanitizer, i.e., it allows the following transitions: RED → REM, RED → PRE.

[837] Namely, the schemes *privRSS*, *flexRSS* and *structRSS* offer some kind of consecutive redaction controls.

The formalisation of the security of this property is given in Sec. 14.9.2.1 in the context of the $flex\mathcal{RSS}$ scheme.

Hence, consecutive redaction control allows a Signer to delegate the Sanitizer the right to remove existing rights to redact or sanitize initially given by the Signer. In doing so the Sanitizer only decreases the future sanitization or redaction possibilities, and thus prevents a consecutive adversarial Sanitizer form carrying out the “additional sanitization attack” [264, 347].

14.5.1.2. Relations between the adapted and existing security properties

The implications given in Sec. 11.14 do not change: transparency $\Rightarrow$ privacy, privacy $\not\Rightarrow$ transparency, and unforgeability is independent. Proofs are omitted as only minor adjustments are required.

14.5.2. $priv\mathcal{RSS}$ algorithmic description for an RSS for trees ($\mathcal{RSS}_T$)

In the following the harmonised algorithmic description of RSS (see Sec. 11.9) is adapted to cater explicitly for trees $T = (\mathcal{V}, \mathcal{E})$ where the redaction operation would allow any node, including the root, to be redacted. As the idea is to keep the Signer in full and explicit control an admissible modification policy is provided, like ADM. Namely, the signing algorithm TSign was expanded with two flags (f_{root}, f_{int}), which control non-leaf redactability. This yields the following description of $\mathcal{RSS}_T$:

Definition 219 : *priv$\mathcal{RSS}$*

The RSS for trees consists of four efficient (PPT) algorithms: $\mathcal{RSS}_T := ($KeyGen, TSign, TVerify, TShare$)$.

Key Generation: *The key generation algorithm on input of a security parameter λ outputs a key pair consisting of the private key sk and the public key pk:*

$$(sk_{sig}, pk_{sig}) \leftarrow \mathsf{KeyGen}(1^\lambda)\,.$$

Signing: *The signing algorithm TSign takes as input the secret key sk, the tree T, a flag $f_{root} \in \{0,1\}$ indicating, if the root is allowed to be redacted, and a flag $f_{int} = \{0,1\}$ which indicates, if intermediary (i.e. non-leaf) nodes can be redacted. By definition a 1 indicates that the corresponding action is allowed. It outputs a signature σ_T over the tree T (or $\perp$ on error):*

$$(T, \sigma_T) \leftarrow \mathsf{TSign}(1^\lambda, sk, T, f_{root}, f_{int})\,.$$

Redacting: *The algorithm TShare takes as input a tree T, a public key pk and signature σ_T, as well as a set of nodes $\mathcal{N} \subseteq T$ to redact. It returns a new tree $T' \leftarrow \mathsf{MOD}(T, \mathcal{N})$, along with a new signature σ'_T resp. $\perp$ on error, where MOD is the function modifying the tree T with respect to $\mathcal{N}$, i.e., $T' = T \setminus \mathcal{N}$. Hence, TShare outputs (or $\perp$ on error):*

$$(T', \sigma'_T) \leftarrow \mathsf{TShare}(1^\lambda, T, pk, \sigma_T, \mathcal{N})\,.$$

$\mathcal{RSS}_T$ does not consider, if and how the edges between the nodes are treated, if a redaction takes place. This must then concern the respective instantiation of a $\mathcal{RSS}_T$. In particular, it depends on the concrete use case, if non-leaf nodes are allowed to be redacted, which is delegated to the Signer for full control.

Verification: *The algorithm TVerify takes as input a tree T, a public key pk and the signature σ_T. It is assumed that the flags f_{root} and f_{int} can be reconstructed from a valid tree-signature pair. It outputs a decision $d \in \{\texttt{false}, \texttt{true}\}$, which indicates, if σ_T is a valid ($d = \texttt{true}$) signature on T under the public key pk:*

$$d \leftarrow \mathsf{TVerify}(1^\lambda, T, pk, \sigma_T)\,.$$

14.5.3. $priv\mathcal{RSS}$ correctness properties

The usual correctness properties for RSS as given in (see Sec. 11.11) must hold.

14.6. $priv\mathcal{RSS}$ cryptographic instantiation

The resulting Construction 6 has been published, as part of the joint work with *K. Samelin*, *A. Bilzhause*, *J. Posegga* and *H. de Meer* [425] at ACNS 2012 (see Appendix A publication n° 10).

14.6.1. Cryptographic preliminaries

The instantiation of $priv\mathcal{RSS}$ requires a secure aggregate signature scheme with public aggregation and a secure and transparent RSS for ordered lists.

14.6.1.1. Aggregate signatures from bilinear pairings

Aggregate signatures ($\mathcal{AGG}$) have been introduced by *Boneh* et al. in [58]. The basic idea is as follows: given ℓ signatures, i.e., $\{\sigma_i \mid 0 < i \leq \ell\}$, a $\mathcal{AGG}$ constructs one compressed signature σ_c which contains all signatures σ_i. This allows verifying all given signatures σ_i by verifying σ_c. To construct such a scheme, let $\mathbb{G}_1$ be a cyclic multiplicative group with prime order q, generated by g, i.e., $\mathbb{G}_1 = \langle g \rangle$. Further, let $\mathbb{G}_T$ denote a cyclic multiplicative group with the same prime order q. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$, where $\mathbb{G}_T = \langle \hat{e}(g,g) \rangle$, be a bilinear map such that:

1. Bilinearity: $\forall u, v \in \mathbb{G}_1 : \forall a, b \in \mathbb{Z}/q\mathbb{Z} : \hat{e}(u^a, v^b) = \hat{e}(u,v)^{ab}$
2. Non-degeneracy: $\exists u, v \in \mathbb{G}_1 : \hat{e}(u,v) \neq 1$
3. Computability: There is an efficient algorithm $\mathcal{A}_{bimap}$ that calculates the mapping $\hat{e}$ for all $u, v \in \mathbb{G}_1$

Definition 220 : The BGLS-Scheme

The $\mathcal{AGG}$ by Boneh *et al. [58] (BGLS-Scheme) with public aggregation consists of five efficient algorithms. In the following one public key, Q, is used. This allows a performance improvement, while making sure that just one signing key is used. It is defined as follows:*

$$\mathcal{AGG} := (\mathsf{AKeyGen}, \mathsf{ASign}, \mathsf{AVerf}, \mathsf{AAgg}, \mathsf{AAggVerf})$$

AKeyGen: *The algorithm* KeyGen *outputs the public and private key of the* Signer*, i.e.,* $(\mathsf{pk}, \mathsf{sk})$. $\mathsf{sk} \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$ *denote the* Signer*'s private key. Additionally, let* $\mathcal{H}_k : \{0,1\}^* \to \mathbb{G}_1$ *be an ordinary cryptographic hash-function from the family* $\mathcal{H}_K$*, modelled as a random oracle, and set* $Q \leftarrow g^{\mathsf{sk}}$*, where* $\langle g \rangle = \mathbb{G}_1$*. Set the public parameters and key* $\mathsf{pk} \leftarrow (g, Q, \mathbb{G}_1, \mathbb{G}_T, \mathcal{H}_k, \hat{e})$.

ASign: *The algorithm* ASign *outputs the signature* σ_i *on input of the secret key* sk *and a single string* $m_i \in \{0,1\}^*$: $\sigma_i \leftarrow (\mathcal{H}_k(m_i))^{\mathsf{sk}}$.

AVerf. *To verify a signature* σ_i*, check, if the following equation holds:* $\hat{e}(\sigma_i, g) \overset{?}{=} \hat{e}(\mathcal{H}_k(m_i), Q)$.

AAgg. *To aggregate* ℓ *signatures* σ_i *into an aggregated signature* σ_c*, the aggregator computes* $\sigma_c \leftarrow \prod_{i=1}^{\ell} \sigma_i$*, denoted as* $\mathsf{AAgg}(\mathsf{pk}, \mathcal{S})$*, where* $\mathcal{S}$ *is the set of* ℓ *signatures signed using the same public parameters contained in* pk*.* AAgg *can be used by untrusted parties and without knowing the private key.*

AAggVerf: *To verify an aggregated signature* σ_c*, check whether* $\hat{e}(\sigma, g) \overset{?}{=} \prod_{i=1}^{\ell} \hat{e}(\mathcal{H}_k(m_i), Q)$ *holds, on input of* σ_c*,* pk *and a list of all signed* m_i*. To improve efficiency, the right side can be rewritten as* $\hat{e}(\prod_{i=1}^{\ell} \mathcal{H}_k(m_i), Q)$*, due to the use of only one public key. In the remainder this is denoted using the algorithm as follows:* $d \leftarrow \mathsf{AAggVerf}(\mathsf{pk}, \sigma, \{m_i\}_{0<i\leq\ell})$.

The usual correctness requirements must hold, which have been formally proven in [58]. Moreover, the expected security properties must hold, i.e., unforgeability under chosen message attacks (UNF-CMA) and the k-element extraction assumption. The proofs and formal definitions can also be found in [58].

14.6.2. $priv\mathcal{RSS}$ construction

The basic idea of $priv\mathcal{RSS}$ is to use the fact that a tree is uniquely determined by its pre- and postorder traversal numbers. However, as *Brzuska* et al. show in [66], ordering random numbers leads to insecure schemes.

$priv\mathcal{RSS}$ builds on the fact that removing an element from a uniformly distributed list of random numbers preserves their distribution. This list (indeed there are actually two lists) is managed by a secure order preserving transparent RSS. The lists are used to encode all nodes' post- and pre-order traversal numbers[838]. Hence, $priv\mathcal{RSS}$ signs random numbers in an order protecting manner by using a secure transparent RSS, which protects the order of the traversal numbers. This allows to randomise the traversal numbers itself without having to explicitly retain the ordering within the range of the random number, i.e. no order-preserving randomisation. Instead, $priv\mathcal{RSS}$ uses the order-preservation offered by the RSS that is used to sign two lists. Each list contains $|V|$ uniformly distributed random nonces, which are pairwise unique. $priv\mathcal{RSS}$ uses one list for pre- and another one for post-order traversal numbers. Each node n_i is annotated with the nonce at the positions in the lists corresponding to the node's traversal numbers. This got nicely depicted in Fig. 109, which is taken from the slides prepared and presented by *K. Samelin* when presenting the joint work at ACNS 2012 which contains the results presented here (see Appendix A publication n° 10).

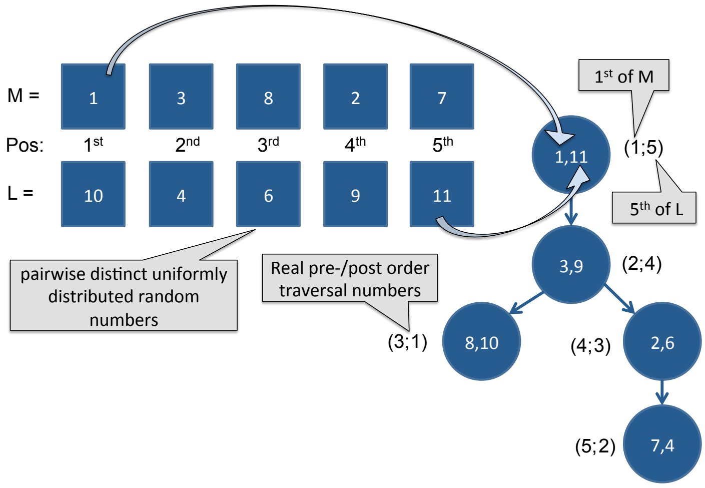


Figure 109. $priv\mathcal{RSS}$'s construction is built upon two ordered lists of pairwise unique (i.e., M and L will not contain the same number) random numbers signed by an order protecting transparent RSS; This figure is taken from the presentation by *K. Samelin* used to present joint work containing $priv\mathcal{RSS}$ at ACNS 2012 [425]

The ordering of each list is secured by signing it with the RSS. Note, the nonces in the lists are not ordered. When a node n_i is redacted, the two nonces used to annotate n_i are removed from the respective list using the RSS, while leaving each list of remaining nonces still uniformly distributed. Additionally, the Signer can decide, if a redaction of the root ($f_{root} = 1$) or redaction of intermediate nodes ($f_{int} = 1$) is allowed, as the Signer annotates the nodes accordingly. This keeps the Signer in control, while giving more flexibility. This must be done by annotating nodes accordingly. One cannot use the content of the nodes in the lists, as they may not be unique, making the reconstruction ambiguous and therefore forgeable. Finally, the aggregate signature is used to improve the performance of the verification process and to achieve consecutive redaction control, as shown in Sec. 14.5.1.1.

[838] The traversal numbers are the position of nodes in a depth-first search of a tree; pre-order first takes the node and then traverses into the left (or right) subtree; post-order first traverses into the left (or right) subtree and then takes the node; see Fig. 109 for an example or [130].

Construction 6 : $priv\mathcal{RSS}$

Let $\mathcal{AGG} := (\mathsf{AKeyGen}, \mathsf{ASign}, \mathsf{AVerf}, \mathsf{AAgg}, \mathsf{AAggVerf})$ be a secure (unforgeability under chosen message attacks (UNF-CMA) and the k-element extraction assumption) aggregate signature scheme with public aggregation[839] using just one key pair.

Let $\Pi := (\mathsf{Redact}$ be a secure and transparent redactable signature scheme for ordered lists. For example the RSS *published as joint work with K. Samelin, J. Posegga and H. de Meer in [392] (see Appendix A publication n^o 24).*

Define $priv\mathcal{RSS} := (\mathsf{KeyGen}, \mathsf{TSign}, \mathsf{TVerify}, \mathsf{TShare})$ as follows:

Key Generation: *The key generation algorithm outputs two key pairs: (1) A key pair for an aggregate signature scheme $\mathcal{AGG}$, i.e., $(sk_{\mathcal{AGG}}, pk_{\mathcal{AGG}}) \leftarrow \mathcal{AGG}.\mathsf{AKeyGen}(1^\lambda)$. (2) A key pair for Π, i.e., $(sk_\Pi, pk_\Pi) \leftarrow \Pi.\mathsf{KeyGen}(1^\lambda)$. Output $(\mathsf{sk}, \mathsf{pk}) = ((\mathsf{sk}_\Pi, \mathsf{sk}_{\mathcal{AGG}}), (\mathsf{pk}_\Pi, \mathsf{pk}_{\mathcal{AGG}}))$*

Signing: *The signing algorithm $(T, \sigma_T) \leftarrow \mathsf{TSign}(\mathsf{sk}, T, f_{root}, f_{int})$ takes all secret keys, the tree T to sign and the flags f_{root} and f_{int}. To sign T, perform:*

1. *Generate two lists, $\mathcal{L}$ and $\mathcal{M}$, each containing $n = |V|$ pairwise distinct uniformly distributed random numbers $\in \{0,1\}^\lambda$. The elements of the list are addressed by an index, i.e., $\mathcal{L}_i$.*
2. *Let i be the index of a pre-order traversal $c_i \leftarrow \mathcal{L}_i||c_i$.*
3. *Let j be the index of a post-order traversal. Set $c_i \leftarrow \mathcal{M}_j||c_i$.*
4. *Choose a random tag $\tau \overset{\$}{\leftarrow} \{0,1\}^\lambda$, which must be unique for each tree T*
5. *If $f_{root} = 0$, set $\tau \leftarrow \tau||noroot$ and annotate the root: $c_r \leftarrow c_r||theroot$, while all for all other nodes: $c_i \leftarrow c_i||nottheroot$. Otherwise, set $\tau \leftarrow \tau||root$ and annotate all nodes: $c_i \leftarrow c_i||nottheroot$. Sign τ: $\sigma_\tau \leftarrow \mathcal{AGG}.\mathsf{ASign}(\mathsf{sk}_{\mathcal{AGG}}, \tau)$.*
6. *Draw a nonce d, $d \overset{\$}{\leftarrow} \{0,1\}^\lambda$. For each node n_i: if $f_{int} = 0$, set $c_i \leftarrow (c_i||d + \mathrm{depth}(n_i))$, otherwise $c_i \leftarrow (c_i|| - 1)$. The function $\mathrm{depth} : V \to \mathbb{N}$ returns the distance from the root to the given node $n_i \in T$.*
7. *For each node n_i sign $c_i||\tau$: $\sigma_i \leftarrow \mathcal{AGG}.\mathsf{ASign}(\mathsf{sk}_{\mathcal{AGG}}, c_i||\tau)$.*
8. *Compress all resulting signatures into the aggregate signature σ_c, along with the signature of τ, i.e., $\sigma_c \leftarrow \mathcal{AGG}.\mathsf{AAgg}(\mathsf{pk}_{\mathcal{AGG}}, \sigma_\tau \cup \{\sigma_i\}_{0<i\leq n})$.*
9. *Sign $\mathcal{L}$ using Π, i.e., $(\mathcal{L}, \sigma_\mathcal{L}) \leftarrow \Pi.\mathsf{Sign}(\mathsf{sk}_\Pi, \mathcal{L})$.*
10. *Sign $\mathcal{M}$ using Π, i.e., $(\mathcal{M}, \sigma_\mathcal{M}) \leftarrow \Pi.\mathsf{Sign}(\mathsf{sk}_\Pi, \mathcal{M})$.*
11. *Output (T, σ_T), where $\sigma_T = (\sigma_c, (\sigma_i)_{0<i\leq n}, \sigma_\tau, \mathcal{L}, \mathcal{M}, \sigma_\mathcal{L}, \sigma_\mathcal{M}, \mathsf{pk}_{\mathcal{AGG}}, \mathsf{pk}_\Pi, \tau)$.*

Redacting: *The algorithm $\mathsf{TShare}(T, \mathsf{pk}, \sigma_T, \mathcal{N})$ takes as input the tree T, all public keys* pk *and the signature σ_T, as well as a set of nodes $\mathcal{N} \subseteq T$.*

1. *Remove nodes by setting $T' \leftarrow T \setminus \mathcal{N}$.*
2. *If intermediate nodes have been redacted, adjust the edges of the intermediate nodes' successors. In particular, for each node $n_i \in T' \setminus r$ not having a parent, introduce the edge $e_{i,j}$, where n_j is the closest ancestor node not redacted. If the result is not a tree, return $\bot$.*
3. *For each $n_i \in \mathcal{N}$, adjust both lists of nonces: $\sigma'_\mathcal{L} \leftarrow \Pi.\mathsf{Redact}(\mathsf{pk}_\Pi, \mathcal{L}, i)$, where i is the pre-order number of n_i. And $\sigma'_\mathcal{M} \leftarrow \Pi.\mathsf{Redact}(\mathsf{pk}_\Pi, \mathcal{M}, j)$, where j is the post-order number of n_i*
4. *Set $\sigma'_c \leftarrow \mathcal{AGG}.\mathsf{AAgg}(\mathsf{pk}_{\mathcal{AGG}}, \sigma_\tau \cup \{\sigma_i\}_{0<i\leq n'}\})$*
5. *Construct (T', σ'_T), where $\sigma'_T = (\sigma'_c, (\sigma_i)_{0<i\leq n'}, \sigma_\tau, \mathcal{L}', \mathcal{M}', \sigma'_\mathcal{L}, \sigma'_\mathcal{M}, \mathsf{pk}_{\mathcal{AGG}}, \mathsf{pk}_\Pi, \tau)$.*

[839] For example the *Boneh* et al. [58] (BGLS-Scheme).

6. *Verify σ_T: If TVerify(T', pk, σ'_T) outputs* `false`*, abort and return $\perp$.*
7. *Output (T', σ'_T).*

Verification: *Verifying the signature is similar to generating the signature.*

1. *Check the validity of $\sigma_{\mathcal{L}} / \mathcal{L}$ and $\sigma_{\mathcal{M}} / \mathcal{M}$ using Π.Verify.*
2. *For each node $n_i \in T$, parse c_i as $(m_i, l_i, e_i, a_i, d_i)$.*
3. *Traverse T via pre-order: $\forall n_i \in T$, check if $\mathcal{L}_i = l_i$. If not, abort and return* `false`*.*
4. *Traverse T via pre-order: $\forall n_j \in T$, check if $\mathcal{M}_j = m_j$. If not, abort and return* `false`*.*
5. *Compute $v \leftarrow \mathcal{AGG}$.AAggVerf$(pk, \sigma_T, \{\tau\} \cup \{m_i||l_i||e_i||d_i||a_i||\tau\}_{0<i\leq n})$. If $v = 0$, abort and return* `false`*.*
6. *Let r denote the root of the tree T. If $d_r \neq -1$, check for all nodes $n_i \in T \setminus r$, if $\text{depth}(\text{parent}(n_i) = d_i - 1)$, where $\text{parent} : V \to V$ returns the only parent of a given node n_i. If not, abort and return* `false`*.*
7. *Parse τ as (τ, h). If $h =$ noroot, check, if the received root's annotation r_a equals "theroot". If so, return* `true`*, otherwise return* `false`*.*

14.6.3. $priv\mathcal{RSS}$ runtime and complexity

The $priv\mathcal{RSS}$ construction 6 has only an overhead of $\mathcal{O}(n)$, if the underlying order-preserving RSS is in $\mathcal{O}(n)$. Such an order-preserving RSS is the RSS published jointly with *K. Samelin*, *J. Posegga* and *H. de Meer* in a technical report [392] (see Appendix A publication n^o 24). Also the storage complexity of $priv\mathcal{RSS}$ is $\mathcal{O}(n)$, where n is the number of vertices in a tree $T = (V, E)$. See Sec. 14.6.3 for details on the runtime and complexity of $priv\mathcal{RSS}$. $priv\mathcal{RSS}$ demonstrates that the underlying idea of using traversal numbers to transparently redact nodes combined with a order-preserving RSS can be facilitated to build a simplistic and enhanceable redactable scheme, which is provably secure[840] in terms of transparency, privacy and unforgeability, while being highly efficient and very flexible.

All other current schemes in $\mathcal{O}(n)$, e.g., the ones introduced in [265] or [296], are susceptible to the attack on privacy introduced by *Brzuska et al.*'s for the scheme by *Kundu and Bertino* in [66] (see Sec. 12.1.2 for attacks on [265] and Sec. 12.1.4 and Sec. 12.2.1.1 for attacks on [296]). This is due to the fact that appending ordered random numbers cannot be sufficient to hide redactions as shown by *Brzuska et al.* [66]. Other provably secure[841] and transparent schemes proposed, e.g., [66], have a complexity of $\mathcal{O}(n^2)$, are only useable for lists [117, 426], or offer less flexibility, i.e., quoting from ordered lists [5]. The worst-case approximation of the scheme introduced by *Brzuska et al.* in [66] depends on the tree's branching factor, which in the case of a tree with height 1, is $\mathcal{O}(n)$, where n is the number of leaf-nodes. As the branching factor is not a constant factor anymore, the growth will be quadratic in n.

In the joint publication [425] (see Appendix A publication n^o 10) the scheme by *Kundu and Bertino* in the original version [295] and the scheme by *Brzuska et al.* [66] have been implemented to demonstrate the impact of being quadratic in n has on the practical usability. Note, the speed of $priv\mathcal{RSS}$ only differs by a constant factor from the scheme by *Kundu and Bertino* in the original version [295], i.e., the underlying RSS Π. The tests were performed on a laptop with an *Intel* T8300 Dual Core @2.40 GHz and 4 GiB of RAM. It ran *Ubuntu* Version 10.04 LTS (64 Bit) and Java version `1.6.0_26-b03`. The security parameter was 2048 Bit for generating RSA-based keys. The trees signed are n-ary balanced ones with height h. Tab. 19 and 20 show the results for high trees with a low branching factors. Tab. 21 and 22 show the results for flat trees with a high branching factor. This gives a good impression for different use cases and allows determining the specific pros and cons of the schemes.

[840] As defined in Sec. 2.3.1.
[841] As defined in Sec. 2.3.1.

	Generate σ_T			Verify σ_T		
h \ n	10	50	100	10	50	100
2	721	14,319	55,625	24	525	1,910
3	6,856	-slow-	-slow-	241	-slow-	-slow-

Table 19. *Brzuska* et al.: Low Branching Factor n; Median Runtime in ms

	Generate σ_T			Verify σ_T		
h \ n	10	50	100	10	50	100
2	544	2,247	5,293	5	8	10
3	4,319	106,694	369,854	8	165	319

Table 20. *Kundu and Bertino*: Low Branching Factor n; Median Runtime in ms

	Generate σ_T			Verify σ_T		
h \ n	2	3	4	2	3	4
5	605	2,804	9,607	19	95	333
10	17,195	666,530	-slow-	614	18,838	-slow-

Table 21. *Brzuska* et al.: High Branching Factor n; Median Runtime in ms

	Generate σ_T			Verify σ_T		
h \ n	2	3	4	2	3	4
5	1,657	5,065	14,132	4	12	22
10	44,756	1,228,738	-slow-	111	864	-slow-

Table 22. *Kundu and Bertino*: High branching factor n; Median Runtime in ms

As shown in Tables 19 to 22, all schemes have a comparable runtime for small trees. For binary and ternary trees of increasing depth the scheme by *Brzuska et al.* suffers from the quadratic runtime and becomes unusable, denoted by “-slow-”, very fast. Measuring was aborted, if the runtime exceeded 20 minutes. This supports the intuition that — as a minimum — a linear complexity is required to yield useable schemes, as waiting a few minutes to generate σ_T is not acceptable in most applications, even if signatures are normally more often verified than generated.

14.6.4. $priv\mathcal{RSS}$ security proof

In the following the proofs of transparency and unforgeability are presented.

Theorem 28 : $priv\mathcal{RSS}$ from Construction 6 is Transparent.

$priv\mathcal{RSS}$ Construction 6 is transparent and therefore also private [66].

Proof 28

$\mathcal{L}$, $\mathcal{M}$, the public keys $\mathsf{pk}_{\mathcal{AGG}}$ and pk_{Π} and the tag τ do not leak any information about the tree T. The lists $\mathcal{L}$ and $\mathcal{M}$ contain uniformly distributed random numbers. Even on redaction of a tree node, the removal only removes elements from a uniformly distributed list of random numbers, which still results in a uniformly distributed list of random numbers. If the tree is annotated with d, nothing can be derived either, since d is chosen from a uniform distribution and the tree grows at most linearly in $|V|$. Hence, one only needs to show that $\sigma_{\mathcal{L}/\mathcal{M}}$ and σ_c together imply transparency. A successful attack on transparency would show that either Π and/or the aggregate signature scheme $\mathcal{AGG}$ is not transparent. The proofs for transparency of σ_c are omitted here and can be found in [348] and [426]. For $\sigma_{\mathcal{L}/\mathcal{M}}$, it remains to show how an adversary would also break the transparency of the underlying $\mathsf{RSS}\,\Pi$: Assume an efficient adversary $\mathcal{A}_{tra}$ which wins the transparency game of the proposed scheme. Using $\mathcal{A}_{tra}$ one constructs another adversary $\mathcal{B}_{tra}$ to break the transparency of Π in the following way: For any calls to $\mathcal{O}^{\mathsf{TSign}}$ resp. $\mathcal{O}^{\mathsf{Adapt/Sign}}$, $\mathcal{B}_{tra}$ genuinely returns the answers of its own oracle. Eventually, $\mathcal{A}_{tra}$ outputs its guess b^*. b^* is then outputted by $\mathcal{A}_{tra}$ as its own guess. It remains to argue that, due to the fact that $\mathcal{AGG}$ is

information-theoretically transparent, $\mathcal{B}_{tra}$'s probability of success equals the one of $\mathcal{A}_{tra}$. Only Π could have leaked this information, as the lists contain just uniformly distributed nonces and redactions of elements in that list again lead to a uniformly distributed list. Hence, $\mathcal{B}_{tra}$ wins with the same probability as $\mathcal{A}_{tra}$. □

Theorem 29 : *priv$\mathcal{RSS}$* from Construction 6 is Unforgeable.

priv$\mathcal{RSS}$ Construction 6 is unforgeable.

Proof 29

Assume that no tag-collisions occur, winning the unforgeability game in a trivial manner. Note, that one does not need an induction over the tree, as it gets transformed into two lists, which are uniquely determined. Let the algorithm winning the unforgeability game be denoted as $\mathcal{A}$. priv$\mathcal{RSS}$ Construction 6's security relies upon the security of $\mathcal{AGG}$ and Π. Given the game for unforgeability Fig. 57 on page 287, one can derive that a forgery must fall in at least one of the following cases:

Case 1: A value protected by σ_c has never been signed by the oracle

Case 2: The value protected by σ_c has been signed, but $T^* \not\subseteq span_{\vdash}(T)$. In other words: The tree T^* protected by σ_T is not in the transitive closure of any tree for which a signature was queried. This case has to be divided as well:

Case 2a: $T \notin span_{\vdash}(T^*)$

Case 2b: $T \in span_{\vdash}(T^*)$

One constructs an adversary $\mathcal{B}$, which breaks the unforgeability of the BGLS-Scheme, if an adversary $\mathcal{A}$ with a non-negligible advantage ϵ exists, winning the unforgeability game. To do so, $\mathcal{B}$ uses $\mathcal{A}$ as a black box. For every signature query $\mathcal{A}$ requests, $\mathcal{B}$ forwards the queries to its signing oracle $\mathcal{O}^{TSign}$ and genuinely returns the answers to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs (T^*, σ_T^*), where $T^* = (\sigma_c^*, \{\sigma_i\}_{0<i\leq n^*}, \sigma_\tau^*, \mathcal{L}^*, \mathcal{M}^*, \sigma_{\mathcal{L}}^*, \sigma_{\mathcal{M}}^*, \mathsf{pk}_{\mathcal{AGG}}^*, \mathsf{pk}_{\Pi}, \tau^*)$. Given the transcript of the simulation, $\mathcal{B}$ checks, if the outputted tuple is a trivial "forgery", i.e., just an allowed redaction. If so, $\mathcal{B}$ aborts the simulation. If $\mathcal{B}$ does not abort, one has to consider the following constellations: If σ_c^* contains messages never signed, $\mathcal{B}$ outputs (σ_c^*) along with forged strings, which can easily be extracted. If $T \notin span_{\vdash}(T^*)$, one has to consider two cases: (1) the values protected by $\sigma_{\mathcal{L}}$ resp. $\sigma_{\mathcal{M}}$ have been altered or (2) the strings protected by σ_T were modified. In either case, this allows to break the unforgeability of either Π or $\mathcal{AGG}$. The corresponding forgeries can easily be extracted. If $T \in span_{\vdash}(T^*)$, both Π and $\mathcal{AGG}$ must have been forged, since new elements are now contained in the aggregate or the RSS. *As before, the forgeries can be extracted given the transcript.* □

14.6.5. *priv$\mathcal{RSS}$* integrity offer and visualisation

priv$\mathcal{RSS}$ is a redactable signature scheme for a tree T. It offers transparency (and therefore privacy). It is an RSS that allows for public redactions. In general, this yields no detection capability for a Verifier for an occurred authorized subsequent modification. Further, it yields no option for the Signer to generate an interactive proof that a message was redacted, i.e., by providing a linking non-redacted variant of the same document. The reason is that *priv$\mathcal{RSS}$* offers no commitment to the signed tree on its own.

However, it could inherit these from the underlying schemes: If the underlying RSS scheme is non-updatable, i.e., offers non-repudiation properties (as defined in Sec. 2.10.2.1) towards the accountability of the Signer, then new nodes can not be added by the Signer after the signature was generated. Also the aggregate signature scheme's properties might be able to be favourable in respect of achieving non-repudiation.

As an ultimate solution *priv$\mathcal{RSS}$* can be paired with an SSS that offers ACA – ≥1CD – PUB integrity following the ARSS framework (see Theorem 44 on page 422). This would make the resulting *priv$\mathcal{RSS}$*-ARSS also Sanitizer-accountable and thus would offer ACA – ≥1CD – PUB integrity.

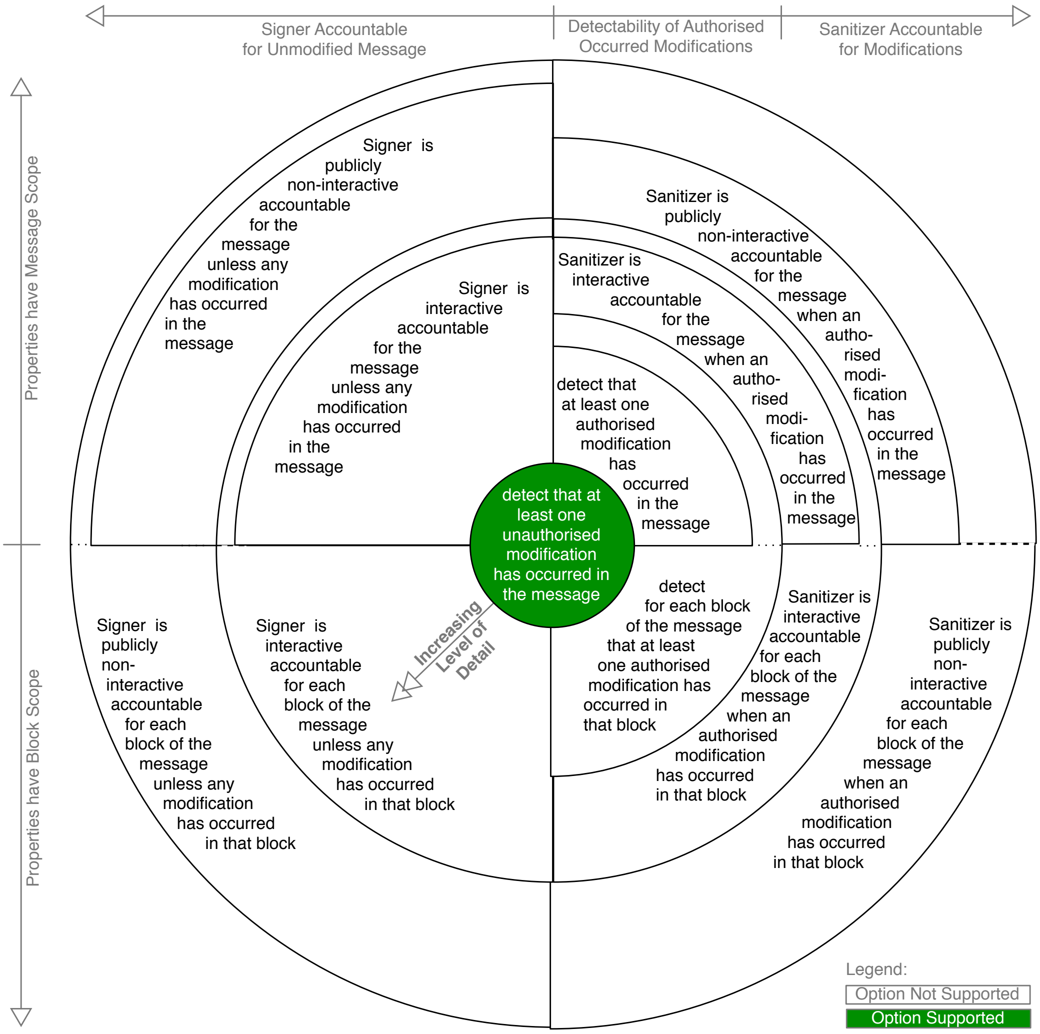

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an authorised modification has occurred in the message
Sanitizer is interactive accountable for the message when an authorised modification has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Increasing Level of Detail
Legend:
Option Not Supported
Option Supported

Figure 110. *privRSS* offers ACA – UCD integrity, because Verifiers can only detect unauthorized subsequent modifications and there is no accountability in *privRSS*; can be added by ARSS

To classify *privRSS*, no further assumptions on the underlying primitives are made and the behaviour based on minimal capabilities of underlying building blocks is taken for classification. Consequently, *privRSS* classifies as giving ACA – UCD integrity with the Signer staying accountable for all valid redactions. As there is no property of linkability a Signer can not prove that a certain redacted message is not original. There are is simply no Judge algorithm in *privRSS*. Fig. 110 visualises this.

14.7. *flexRSS* concept: RSS for trees with flexible intermediate node redaction with Signer control

Like the previous *privRSS*, also *flexRSS*[842] is geared to directly treat tree-based structures. This is helpful for more structured formats, like XML, as the scheme can directly protect the content and structural integrity of the document without needing a mapping.

[842] This result has been published, as part of the joint work with *K. Samelin*, *J. Posegga* and *H. de Meer* [394] at SECRYPT 2012 (see Appendix A publication nº 12) and also in the Journal of E-Business and Telecommunications [134] (see Appendix A publication nº 2).

The $flex\mathcal{RSS}$ scheme was devised to offer standard privacy (see Sec. 11.13.3), i.e., be as secure as *Brzuska et al.*'s first rigid model for secure RSS for trees [66]. However, the model is restrictive with the respect, that it only allows redacting leaves of the tree [66]. In the following Sec. 14.7.2 the harmonised algorithms given in Definition 169 on page 284 get refined to securely allow the possibility to redact any node. This flexibility is an answer to Shortcoming 5[843] as it brings intermediate node redaction and explicit control over the needed re-locations that become necessary after any intermediate node redaction and does also treat the tree's root as a redactable node. Existing schemes that privately allow redaction of intermediate nodes — and also the previous $priv\mathcal{RSS}$ — had to relax[844] the structural integrity protection to the transitive closure of the trees. Note, redacting the root, as shown in Fig. 111 (3c) on page 389, is possible. In the example, the node n_2 from Fig. 111 (3c) transparently becomes the new root. In general, redacting the root might leave the Sanitizer with a forest of trees, i.e., two or more unconnected subtrees. Each subtree is then a valid signed subtree, however the original connection between the subtrees is lost. In other words such a redaction creates several signed trees. In $flex\mathcal{RSS}$ a redaction of the root or of intermediary non-leaf nodes is considered authorized only if the Signer provided all needed re-location edges, such that after an allowed re-location all subtrees can be connected into one tree.

Additionally, $flex\mathcal{RSS}$ allows the Signer to authorize re-location without redaction. This flexibility might also be too much freedom for an application domain. Therefore, $flex\mathcal{RSS}$ allows the Signer to fully control how the Sanitizer can violate the structural integrity protection to keep intermediate or root node redactions private. For example, the flexibility would allow the Signer to explicitly authorize a flattening of hierarchies or other examples given in Sec. 12.5.2.

Consequently, the security requirements need to additionally capture the Signer's flexibility to allow redaction of any node. The proposed framework explicitly takes level promotions due to *re-locations* of specified subtrees into account, which resembles the *implicit* possibility of previous schemes (e.g., present in [295, 296]). In particular, the signing algorithm adds additional edges to the tree to allow future re-locations. A verifying party needs to decide which edge to use depending on the the tree it is presented. This allows the Sanitizer to maintain transparency after occurred modifications. On the other hand, the Signer remains in charge when describing how an occurred redaction is hidden by re-locating the subtree. It enables the Signer to *explicitly* prohibit the redaction of nodes *individually*, as the Signer must *explicitly* sign an edge for re-location. Without that edge the sub-tree below that node would be dangling and thus one is not authorized to redact this node. Additionally, in Construction 7 (see Sec. 14.8.2) the Signer controls the protection of the order of siblings.

The re-location definition of $flex\mathcal{RSS}$ does not require to redact the intermediate node. Further, $flex\mathcal{RSS}$ is defined to work on ordered and un-ordered trees. The redaction of any node includes the ability to authorize the tree's root to be redactable.

14.7.1. $flex\mathcal{RSS}$ extended security properties

The redactable signature scheme $flex\mathcal{RSS}$ achieves the following security properties (**bold** indicates a new security property introduced in this thesis):

- Unforgeability according to Definition 172, and
- Standard Privacy following *Brzuska et al.* according to Definition 174, and
- Transparency according to Definition 175.

Additionally it offers the increased flexibility to

- let the Signer authorize **redaction of intermediary nodes**, i.e. non-leaf nodes, and
- let the Signer authorize **fully flexible re-location of a subtree**, i.e. not limited to the transitive closure and independent of a node-redaction, and
- let the Sanitizer exercise consecutive re-location control, i.e. the Sanitizer can remove an existing right for relocation from the set of rights for consecutive Sanitizers.

[843] Shortcoming 5: Limited control of the structural integrity protection (see Sec. 12.5).

[844] As given in Sec. 12.2.1.1, the protection of transitive closures, i.e., ancestor-descendant relations, is a weaker structural integrity protection and it allows for semantic attacks, e.g. level promotions, as they go unnoticed by the weaker structural integrity protection.

Apart from achieving the security goals, $flex\mathcal{RSS}$ does offer new flexibility. Namely, $flex\mathcal{RSS}$ allows the Signer to explicitly specify all allowed re-locations of a subtree. If a non-leaf is subject to redaction, all subtrees of the node need to be relocated to preserve the tree structure and not get a forest. Otherwise, the dangling subtree will leak knowledge that a redaction took place to the adversary. Hence, specifying at least one re-location for a subtree is a necessity if the subtree's root is an intermediate node which is authorized to be redacted in order to allow the Sanitizer to remove this non-leaf node and preserve privacy. With $flex\mathcal{RSS}$ the decision if a subtree can be re-located at all and the re-location target remains under the sole control of the Signer. It is also fully flexible, i.e. the potential edges can be used by the Signer to authorize a re-location of a subtree to any location in the tree and also independent of a redaction. This flexibility comes at the cost that a Sanitizer can redact a re-location edge and thus prohibit further re-locations by consecutive Sanitizers, this could be seen equal to consecutive sanitization control as introduced by *Miyazaki et al.* [347] (see also Subsections 14.5.1.1 and 14.9.2.1).

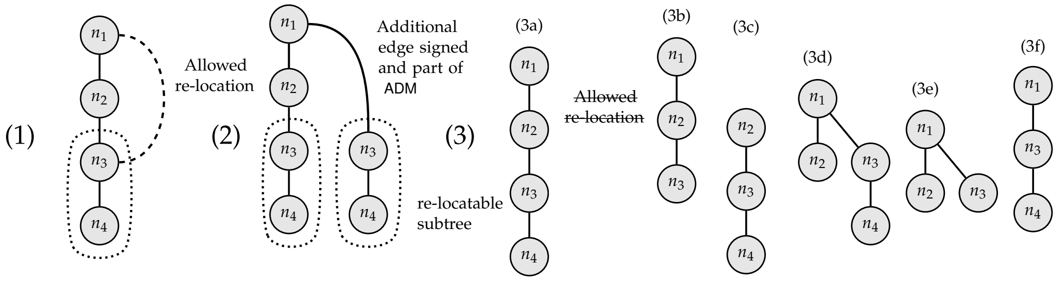


Figure 111. (1) Original unordered tree with allowed non-leaf redaction of n_2 and allowed Level-Promotion of n_3, (2) Expanded tree with duplicates for all possible re-locations of subtrees under n_2, (3) Examples of valid trees with 3 or 4 nodes after: (3a) redact re-locatability, (3b) redact n_4, (3c) redact root node n_1, (3d) relocate-only n_3, (3e) re-locate n_3 and redact n_4, (3f) redact n_2 and level-promote n_3

14.7.1.1. Consecutive intermediate node redaction control by removing re-locate edges

Note, in $flex\mathcal{RSS}$ a third party can prohibit consecutive re-locations by removing the associated witnesses from distribution. An example of a resulting tree is depicted in Fig. 111 (3b). It no longer contains the information of the tree from Fig. 111 (1), which would allow a relocation of n_3 as a direct child of n_1 as depicted in Fig.111 (2). The algorithm Modify from below actually works by removing only the duplicated nodes that have been generated to allow the relocation, and hence it removes the allowed edge $e_{1,3}$. With it the possibility for an intermediate removal of is redacted, as there would be no valid edges to re-locate the dangling subtree after an intermediate node redaction. A full subtree redaction, that would completely remove all nodes of the dangling subtree would still be possible.

14.7.2. $flex\mathcal{RSS}$ algorithmic description

The following notation for trees is used: The root, denoted n_1, is the only node without a parent. Nodes are addressed by n_i. The notation c_i refer to the content of node n_i, which is all the additional information that might be associated with a node, i.e., data, element name and so forth.

Relocation of intermediate nodes is possible with $flex\mathcal{RSS}$. The algorithms of $flex\mathcal{RSS}$ behave in general like the harmonised algorithms given in Section 169. The enhancement of $flex\mathcal{RSS}$ lies in the added, more flexible redact algorithm called Modify. As with regular RSS, running Modify multiple times is the same as providing a MOD that contains more than one change to be made. The output is: $(T', \sigma'_T, \mathsf{ADM}') \leftarrow \mathsf{Modify}(\mathsf{pk}, T, \sigma_T, \mathsf{ADM}, \mathsf{MOD})$. The Modify algorithm of $flex\mathcal{RSS}$ executes redactions as normal for leaf-nodes, but as a bonus allows

- redaction of intermediate, i.e., non-leaf, nodes and then choose where to re-locate the subtree to, and
- redaction of the possibility to re-locate by redacting the 'duplica nodes' and the edges added for re-location.

Hence, prohibiting further *re-locations* is allowed in the $flex\mathcal{RSS}$ scheme. The Signer will always explicitly denote this in ADM, which is explicitly stated in the algorithms of $flex\mathcal{RSS}$ for clarity.

In the following the harmonised algorithms given in Definition 169 are re-defined to securely allow the possibility to redact any node.

Definition 221 : $flex\mathcal{RSS}$: Flexible Redactable Signature Scheme for Trees

The $flex\mathcal{RSS}$ for trees is a an RSS consisting of the following four efficient (PPT)[845] *algorithms: $flex\mathcal{RSS} := (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Modify})$. All algorithms may output $\perp$ in case of an error.*

Key Generation: *The algorithm KeyGen outputs the public and private key of the Signer:*

$$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) .$$

λ denotes the security parameter.

Signing: *On input of the secret key sk, the tree T and ADM the algorithm Sign outputs the signature σ_T. ADM controls what changes by Modify are admissible. In detail, ADM is implemented as a set containing all signed edges, including the ones where a subtree can be re-located to. In particular, $\mathsf{ADM}_T((n_i, n_j)) = 1$ if and only if the edge (n_i, n_j) is to be signed. These edges cannot be derived from T alone. For simplicity, one assumes that ADM is always correctly derivable from (T, σ_T). The output:*

$$(T, \sigma_T, \mathsf{ADM}) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, T, \mathsf{ADM}) .$$

Redacting: *The algorithm Modify takes as input the Signer's public key pk, the tree T, the signature σ_T and ADM of T, and an instruction MOD. MOD contains the actual change to be made: redact a leaf n_i, relocate a subtree T_ψ, distribute a subtree T_v without the original root, or prohibit re-locating a subtree T_ω. A modification of T with respect to MOD is denoted as $T \otimes \mathsf{MOD}$.*

Apart from potentially changing T, the old ADM must be adjusted: In particular, if a node n_i is redacted, the edge to its father needs to be removed from ADM as well. Moreover, if there exists a subtree which could be re-located to become a direct child of the to-be-redacted node, the corresponding edges need to be removed from ADM as well. A modification of ADM with respect to MOD will be denoted as $\mathsf{ADM} \otimes \mathsf{MOD}$. The alteration of ADM is crucial to maintain privacy and transparency.

Verification: *On input of the public key pk, the tree T and the signature σ_T the algorithm Verify outputs a decision $d \in \{\texttt{true}, \texttt{false}\}$ indicating the correctness of the signature σ_T, with respect to pk, protecting the tree T. The output:*

$$d \leftarrow \mathsf{Verify}(1^\lambda \mathsf{pk}, T, \sigma_T) .$$

14.7.3. $flex\mathcal{RSS}$ correctness properties

As in Section 11.11 the correctness of $flex\mathcal{RSS}$ is assumed. Genuinely signed or signed and modified trees are considered valid by the verification algorithm. The following sections use the notation of $span_{\vdash}(T, \mathsf{ADM})$ to denote all valid trees that can be generated from a signed tree T by running Modify never, once or more times, with a MOD which respects ADM (see Definition 59). Hence, all elements of $span_{\vdash}(T, \mathsf{ADM})$ have valid signatures, if and only if T has a valid signature.

14.8. $flex\mathcal{RSS}$ cryptographic instantiation

This result has been published, as part of the joint work with *K. Samelin* and *J. Posegga* and *H. de Meer* [394], at SECRYPT 2012 (see Appendix A publication n° 12) and also in the Journal of E-Business and Telecommunications [134] (see Appendix A publication n° 2).

[845] probabilistic polynomial-time (PPT)

$flex\mathcal{RSS}$ can be instantiated securely based upon a collision-resistant one-way accumulator [24, 43] in combination with the *Merkle*-Hash-Tree-Technique [338]. Employing the *Merkle*-Hash-Tree-Technique enforces the protection of structural integrity. Hence, the $flex\mathcal{RSS}$ scheme can be instantiated using only standard cryptographic primitives.

14.8.1. Cryptographic preliminaries

$flex\mathcal{RSS}$ builds upon the concept of a Merkle hash tree and an accumulating hash-function.

14.8.1.1. Merkle hash tree ($\mathcal{MH}$)

The following construction describes the ideas of the *Merkle* hash tree [338]: The following notation is used for the recursively constructed extended version, which works on arbitrary trees with content. A reversible concatenation of two strings x, y is denoted with $x||y$. The extended *Merkle*-Hash $\mathcal{MH}$ is calculated as: $\mathcal{MH}(x) = \mathcal{H}(\mathcal{H}(c_x)||\mathcal{MH}(x_1)||\ldots||\mathcal{MH}(x_n))$, where $\mathcal{H}$ is a cryptographic hash-function like SHA-512, c_x the content of the node x, x_i a child of x, and n the number of children of x. $\mathcal{MH}(n_1)$'s output, called root-hash-value, depends on all nodes and on the *right order* of the siblings. Hence, signing the root-hash-value protects the integrity of the nodes in an ordered tree and the tree's structural integrity. Obviously, this technique does not allow to hash unordered trees; an altered order will most likely cause a different hash-value for the differently concatenated hash-values of the children. One may argue that annotating an unordered subtree is sufficient. However, this does not allow to re-arrange items and hence enforces a given order, which may not be wanted in certain use-cases, e.g., if invoices are hashed after they are generated by the system the order in the tree would correspond to the order in which they are generated and thus leak the order. Hence, an unsorted list or tree should be signed and verified as such. A more detailed analysis of the *Merkle* hash tree is given in [296], which also gives an introduction to possible inference attacks which break the property of privacy of certain existing schemes.

14.8.1.2. Accumulating hash-functions ($\mathcal{AH}$)

One-way accumulators have been introduced by in [43]. The basic idea is to construct a strongly one-way family of functions $\mathcal{AH}_K$, where $\forall \mathcal{AH}_k \in \mathcal{AH}_K : \mathcal{X}_k \times \mathcal{Y}_k \to \mathcal{X}_k$. In the following just the basic operations of an accumulator are described in compacted notation[846], e.g., no dynamic updates or revocation techniques are required.

Definition 222 : Basic Accumulator $\mathcal{AH}$

A basic accumulator consists of three efficient algorithms, i.e., $\mathcal{AH} := \{\mathsf{KeyGen}, \mathsf{Hash}, \mathsf{Check}\}$:

Key Generation: *The algorithm KeyGen outputs the public system parameters parm on input of a security parameter λ:*

$$\mathsf{parm} \leftarrow \mathsf{KeyGen}(1^\lambda)\,.$$

Digest: *The algorithm Hash on input of a set of to-be-digested items $\mathcal{I} = \{v_1, \ldots, v_n\}$ and the parameters parm outputs the accumulated digest d along with the set of witnesses $\mathcal{W} = \{w_1, \ldots, w_n\}$:*

$$(d, \mathcal{W}) \leftarrow \mathsf{Hash}(\mathsf{parm}, \mathcal{I})\,.$$

The accumulation of $\mathcal{I} = \{v_1; \ldots; v_n\}$ is denoted as $\mathcal{AH}_k(\mathsf{parm}, \{v_1; \ldots; v_n\})$, where $k \in K$. Each witnesses w_ℓ in $\mathcal{W}$ allows to prove the membership of the corresponding value v_ℓ, with $1 \leq \ell \leq n$.

Verification: *The algorithm Check outputs a bit $b \in \{0, 1\}$ indicating if a given value v_i was accumulated into the digest d with respect to prm and a witness w_i. In particular:*

$$b \leftarrow \mathsf{Check}(\mathsf{parm}, v_i, d, w_i)\,.$$

[846] Sometime the witness generation is done by an extra algorithm. Here, its inside the Hash to shorten the interface description.

```
Experiment $\mathsf{Indistinguishable}_{\mathcal{A}}^{\mathcal{AH}}(\lambda)$
    $\mathsf{parm} \leftarrow \mathsf{KeyGen}(1^{\lambda})$
    $b \xleftarrow{\$} \{0,1\}$
    $d^* \leftarrow \mathcal{A}^{\mathsf{LoRHash}(\ldots,b,\mathsf{pk})}(\omega, \mathsf{parm}, \mathsf{pk})$
        where oracle LoRHash for input $\mathcal{S}, \mathcal{R}$:
        $z \xleftarrow{\$} \mathcal{Y}_{\mathsf{pk}}$
        if $b = 1$: return $(\mathsf{Hash}(\mathsf{pk}, \mathcal{S} \cup \mathcal{R} \cup z),$
        $\{(y_i, p_i) \mid p_i \leftarrow \mathsf{Proof}(\mathsf{pk}, y_i \in \mathcal{S}, \mathcal{S} \cup \mathcal{R} \cup z)\})$
        if $b = 0$: return $(\mathsf{Hash}(\mathsf{pk}, \mathcal{S} \cup z),$
        $\{(y_i, p_i) \mid p_i \leftarrow \mathsf{Proof}(\mathsf{pk}, y_i \in \mathcal{S}, \mathcal{S} \cup z)\})$
    return 1, if $d = b$
```

Figure 112. Indistinguishable Experiment for Accumulators

This thesis requires the usual soundness requirements as given by *Barić and Pfitzmann* [24] to hold, while the concrete instantiation of an accumulator must be strongly one-way [24]. In particular, an outsider should not be able to guess members or to generate membership proofs.

Additional information about accumulators can be found in [24, 43, 104]. Moreover, *Derler, Hanser, and Slamanig* [141] offer an overview and a classification of different cryptographic accumulators and their properties.

To achieve transparency, it is also required that the accumulator does not leak how many additional members a digest has. The formal descriptions of these requirements are given below. To maintain transparency, it is required that any output of $\mathcal{AH}$ is distributed uniformly over the co-domain of the hash-function. An accumulator not fulfilling these requirements has been proposed by *Nyberg* in [359]; the underlying *Bloom*-Filter can be attacked by probabilistic methods and therefore leaks the amount of members. This is not acceptable for the use in this thesis. To prohibit recalculations of a digest, this thesis requires a nonce as the seed as already proposed by *Nyberg* [359] and *Barić and Pfitzmann* [24]. The idea to use accumulating hash-functions has already been proposed by *Kundu and Bertino* in [295]. However, *Kundu and Bertino* state that accumulators are not able to achieve the desired functionality. The construction to follow in Sec. 14.8.2 on page 393 will show that they are indeed sufficient.

Collision-Resistance and One-Wayness. The family $\mathcal{AH}_K$ contains only collision-resistant functions [24]. Furthermore, it must be hard to find a digest with the same value without having the pre-images, i.e., require strong one-wayness [24]. Both are captured as follows:

$$\begin{aligned}\Pr[&k \xleftarrow{\$} K; x \xleftarrow{\$} \mathcal{X}_k; y \xleftarrow{\$} \mathcal{Y}_k; (x', y') \leftarrow \mathcal{A}(x, y) : \\ &\mathcal{AH}_k(x, y) = \mathcal{AH}_k(x', y') \wedge y \neq y'] < \epsilon(\lambda)\end{aligned}$$

Where the probability is taken over all coin tosses. In other words, an adversary should not be able to reverse the hashing step and to find a valid preimage or to find any other collision.

Indistinguishability of output. This requires that an adversary cannot decide how many additional members have been digested.

Definition 223

> *Indistinguishability of Accumulator's Output An accumulator is* ***indistinguishable****, if and only if the probability that the game* $\mathsf{Indistinguishable}_{\mathcal{A}}^{\mathcal{AH}}(\lambda)$ *depicted in Fig. 112 returns* 1*, is negligibly close to* $\frac{1}{2}$ *(as a function of* λ*).*

Here, the adversary can choose the input, and has to guess, if the digest has only one more member $(b = 0)$ or more $(b = 1)$. The blinding value z is chosen by the oracle at random. The basic idea is to require that one additionally accumulated blinding value is enough to hide that an accumulator has additional members. Please note that the witnesses are also returned.

Stateless, no dependence on previous input. The accumulating hash-function must be deterministic and should not be able to memorise previous input. It should not have an internal state, allowing it to remember previous input. It should solely depend on the input and the auxiliary information aux.

14.8.2. $flex\mathcal{RSS}$ construction

The underlying idea of the $flex\mathcal{RSS}$ construction is that instead of implicitly allowing the transitive closure to be always the fallback for dangling nodes after redaction of an intermediate node, $flex\mathcal{RSS}$ makes the Signer explicitly define additional edges. This is not limited to edges that are part of the transitive closure. To have these additional edges, $flex\mathcal{RSS}$ requires the Signer to replicate all re-locatable nodes and the underlying subtrees to all the locations where a Sanitizer is later allowed to relocate the node and with it the subtree to. These replicas are generated by the Expand algorithm. The replicas are implicitly used just to produce the re-locatable edges. All algorithms work on nodes not on edges, hence always imagine one builds all allowed re-locations by replicating nodes before one runs an algorithm like Modify, as depicted in Fig. 111 (2). Each additional edge is also noted in ADM. Fig. 111 (1) and 111 (2) visualise this with a dotted area that corresponds to the subtree which contains n_3 and n_4 under the re-locatable n_3. The subtree containing n_3 and n_4 must be re-located as a whole. The dashed curved edge $e_{1,3}$ corresponds to such an additional edge contained in ADM to indicate the allowed re-location of n_3 as direct child of n_1. The accumulator is then used to store a node's contents — combined with a node-individual random seed[847] — and all the node's children. This is then used in a Merkle hash tree (MHT). To redact nodes the Sanitizer removes the redacted nodes from the list of accumulated elements and no longer provides the corresponding witnesses for the redacted elements. During verification the algorithm verifies that for each node x if x's content, $x's$ children and x's order to other siblings is contained in x's *Merkle*-Hash. To sign the ordering between siblings, $flex\mathcal{RSS}$ signs the "left-of" relation, as already used and proposed in [66], [117] and used in [426].

Moreover, the flexibility of $flex\mathcal{RSS}$ to redact any node of the tree does include in its generality the tree's root. Without loss of security, the Signer can add an annotation to the root to prohibit redacting the root by adapting the signing and verification algorithms. During signing one annotates n_1 as root and additionally indicates that redacting the root is prohibited. Complementary, the verify algorithm will check for that indicator and, if and only if present, it will verify, if the annotation is present in the root node of the tree under verification.

In the following it is assumed that witnesses are generated and then distributed as part of the signature. Hence, the Verifier can use them to verify memberships, and, of course, the Sanitizer can remove them from distribution, if the corresponding values are removed. This does not introduce any security problems, since the Sanitizer could give away the original tree anyway. How the witnesses are generated depends on the actual accumulator used and is therefore omitted.

The accumulator also allows $flex\mathcal{RSS}$ to sign unordered trees; for the accumulator it does not matter in what order the members are checked. However, if the structural integrity of ordered trees has to be protected, the ordering between siblings has to be explicitly signed. To allow distribution of subtrees without the root, $flex\mathcal{RSS}$ signs each node's *Merkle*-Hash individually using a standard signature scheme. As said, redaction is therefore a simple removal of the nodes and the corresponding witnesses. Re-location is similar: Apply the necessary changes to T. Additionally, a Sanitizer can prohibit consecutive re-locations, this could be seen equal to consecutive sanitization control [347]. To prohibit further re-location one removes the corresponding witnesses. This implies that ADM is adjusted accordingly.

Construction 7 : $flex\mathcal{RSS}$

Let $\mathcal{AH} := \{\mathsf{KeyGen}, \mathsf{Hash}, \mathsf{Check}\}$ be a secure (collision-resistant, one-way, stateless, with indistinguishable output) accumulating hash-function.

Define $flex\mathcal{RSS} := (\mathsf{KeyGen}, \mathsf{Expand}, \mathsf{Sign}, \mathsf{Modify}, \mathsf{Verify})$ as follows:

Key Generation: *The algorithm KeyGen on input of λ works as follows:*

```
choose $\mathcal{AH}_k \in \mathcal{AH}_K$    //Choose a suitable accumulator $\mathcal{AH}$
set $prm \leftarrow \mathsf{AHKeyGen}_k(\lambda)$
choose $\Pi$                 //Choose an unforgeable signature scheme $\Pi$
set $(pk_S, sk_S) \leftarrow \Pi.A.\mathsf{KeyGen}(\lambda)$
return $(pk = (pk_S, prm, \mathcal{AH}_k, \Pi), sk = (sk_S))$ //Return all generated material
```

Expand: *The algorithm Expand on the input of the tree T and the admissible modification policy (ADM) works as follows:*

[847] The seed is used to prohibit simple recalculation attacks [24].

```
For all edges $e_i \in$ ADM
    replicate the subtree underneath the
        node addressed by $e_i$
        to the designated position
        this must be done bottom-up

Note: This implies that $r_i$ are copied as well.
return this expanded tree, denoted as $\Omega$
```

Signing: *The algorithm* Sign *on input of the secret signature key* sk *and the tree* T *and the admissible modification policy (*ADM*) works as follows:*

```
for each node $n_i \in T$:
    $r_i \xleftarrow{\$} \{0,1\}^\lambda$
    if $r_i$ has already been drawn, draw again
expanded tree $\Omega \leftarrow \mathsf{Expand}(T, \mathsf{ADM})$
for each node $x \in \Omega$:
    draw a random seed $s_x \xleftarrow{\$} \{0,1\}^\lambda$
    let $x_i$ denote a children of $x$
    if the tree is un-ordered:
        $d_x = \mathcal{MH}(x) \leftarrow \mathcal{AH}_k(prm, \{s_x; c_x||r_x;$
            $\mathcal{MH}(x_1); \dots; \mathcal{MH}(x_n)\})$
        $\sigma_x \leftarrow \Pi.\mathsf{A.Sign}(\mathsf{sk}_S, d_x||unordered)$
    else: //ordered tree
        $d_x = \mathcal{MH}(x) \leftarrow \mathcal{AH}_k(prm, \{s_x; c_x||r_x;$
            $\mathcal{MH}(x_1); \dots; \mathcal{MH}(x_n); \Phi_x\})$
        // Build $\Phi_x$, the set of all "left-of" relations of $x_i$:
        $\Phi_x = \{r_i||r_j \mid 0 < i < j \leq n\}$
        $\sigma_x \leftarrow \Pi.\mathsf{A.Sign}(\mathsf{sk}_S, d_x||ordered)$
// Build list of all witnesses:
let $\mathcal{W} = \{w_i | n_i \in \Omega\}$.
return $\sigma_T = (\{\sigma_i\}_{n_i \in \Omega}, \mathcal{W}, \mathsf{ADM})$
```

Redaction: *The algorithm* Modify *on input of public verification key* pk*, the tree* T*, the signature* σ_T*, the admissible modification policy (*ADM*) and the intended redaction instruction (*MOD*) works as follows:*

```
use Verify to verify the tree $T$
expanded tree $\Omega \leftarrow$ Expand(T, ADM)
Case 1: Redact node $n_l$:
    //1. remove **all** $n_l$ (including replicas) from $\Omega$:
    set $\Omega' \leftarrow \Omega \setminus n_l$
    //2. remove **the** node $n_l$ from $T$:
    set $T' \leftarrow T \setminus n_l$
    return $\sigma'_T = (T', \{\sigma_i\}_{n_i \in \Omega'}, \{w_i\}_{n_i \in \Omega'}, \mathsf{ADM})$
Case 2: Share subtree $T_v$, where $n_1 \notin T_v$:
    return $\sigma'_T = (T_v, \{\sigma_i\}_{n_i \in T_v}, \{w_i\}_{n_i \in T_v}, \mathsf{ADM})$
Case 3: Re-locate $T_\psi$:
    set $T' \leftarrow \mathsf{MOD}(T)$
    return $\sigma'_T = (T', \{\sigma_i\}_{n_i \in \Omega}, \{w_i\}_{n_i \in \Omega}, \mathsf{ADM})$
Case 4: Remove re-location edge $e_l$:
    set $\mathsf{ADM}' \leftarrow \mathsf{ADM} \setminus e_l$
    expanded tree $\Omega' \leftarrow \mathsf{Expand}(T, \mathsf{ADM}')$ //Note: Expansion done with $\mathsf{ADM}'$.
    return $\sigma'_T = (T, \{\sigma_i\}_{n_i \in \Omega'}, \{w_i\}_{n_i \in \Omega'}, \mathsf{ADM}')$
```

Verification: *The algorithm* Verify *on input of public verification key* pk*, the tree* T *and the signature* σ_T *works as follows:*

```
check if each $r_i \in T$ is unique.
for each node $x \in T$:
    let the value protected by $\sigma_x$ be $d_x$
```

```
        let $d \leftarrow \textsf{Check}(prm, c_x||r_x, d_x, w_x)$
        if $d = 0$, return $false$
        for all children $c_i$ of $c$ do:
            let the value protected by $\sigma_{c_i}$ be $d_c$
            //Note: checks if children are signed
            let $d \leftarrow \textsf{Check}(prm, d_c, d_x)$
            if $d = 0$, return false
            if $\sigma_x = d_x||ordered$:
                //Is every left-of-relation signed?
                //Note: just linearly many checks
                for all $0 < i < n$:
                    $d \leftarrow \textsf{Check}(prm, r_i||r_{i+1}, d_x, w_{x,x+1})$
                    if $d = 0$, return false
    return true
```

14.8.3. $flex\mathcal{RSS}$ runtime and complexity

For the joint publication with *K. Samelin* and *J. Posegga* and *H. de Meer* [394] at SECRYPT 2012 (see Appendix A publication n^{o} 12) the $flex\mathcal{RSS}$ scheme was implemented and measured.

For generation of σ_T, the sibling's order requires $\frac{n(n-1)}{2}$ hashing steps. All other steps require to touch the nodes only once. Consequently, the runtime approximation of $flex\mathcal{RSS}$'s signing algorithm is linear in the number of nodes, while being quadratic in the number of siblings. Redacting is just removing values. Verification is $\mathcal{O}(|V|)$, since a Verifier has to check, if each digest is contained in its parent's digest and if the content has been digested. Checking the order of siblings can be done in linear time due to the transitive behaviour.

The accumulator is the original construction [43]. Tests were performed on a laptop with an *Intel* T8300 Dual Core @2.40 GHz and 4 GiB of RAM. The OS was *Ubuntu* Version 10.04 LTS (64 Bit) with Java-Framework `1.6.0_26-b03` (OpenJDK). Trees with unordered siblings and one with ordered siblings were measured. Time for generation of keys for the hash-function is included. However, the time for creating the required key pairs is excluded. For the digest calculation all intermediate result are stored in RAM to avoid including any disk access overhead in the timings.

	Generation of σ			Verification of σ		
Nodes	10	100	1,000	10	100	1,000
Ordered	276	6,715	57,691	26	251	2,572
Unordered	103	599	5,527	21	188	1,820
SHA-512	4	13	40	4	13	40

Table 23. $flex\mathcal{RSS}$ Performance; median runtime in ms

As shown in Tab. 23, the $flex\mathcal{RSS}$ construction is useable, but becomes slow for large branching factors. The advanced features come at a price; $flex\mathcal{RSS}$ is considerably slower than a standard hash-function like SHA-512. Note, in many application scenarios signatures are more often verified than generated, so the overhead for verification has a greater impact. All other provable secure and transparent schemes, i.e., [66] and [117], have the same complexity and therefore just differ by a constant factor, but [66] and [117] do not provide a performance analysis on real data.

14.8.4. $flex\mathcal{RSS}$ security proof

$flex\mathcal{RSS}$ shown in Construction 7 is unforgeable, private and transparent. Assuming $\mathcal{AH}$ is strongly one-way, and the signature scheme Π is UNF-CMA,$flex\mathcal{RSS}$ Construction 7 is unforgeable, while $\mathcal{AH}$ always outputs uniformly distributed digests and witnesses, the scheme is transparent and also private. Note, it is enough to show that Transparency and Unforgeability hold because *transparency* $\Rightarrow$ *privacy* [66].

The changes to the security model do not affect the implications and separations as presented in (see Sec. 11.14). Hence, unforgeability is independent, while *transparency* $\Rightarrow$ *privacy* and *privacy* $\not\Leftarrow$ *transparency* [66]. The proofs are essentially the same as already given by *Brzuska et al.* [66]. Following from that it is sufficient to show that transparency and unforgeability hold to show that the scheme is secure. For readability each property is proven on its own.

One the one hand, $flex\mathcal{RSS}$ allows to re-locate a re-locatable subtree without redaction of nodes. On the other hand, $flex\mathcal{RSS}$ prohibits simple copy attacks, i.e., leaving a relocated subtree T_ω in two locations, because each node n_i gets an associated unique nonce r_i. The whole tree gets signed as in the standard *Merkle*-Hash-Tree, with one notable exception: Instead of using a standard hash like SHA-512, $flex\mathcal{RSS}$ deploys an accumulator (see Sec. 14.8.1.2) which allows the Sanitizer to remove values without changing the digest value.

Theorem 30 : $flex\mathcal{RSS}$ from Construction 7 is Unforgeable.

> *If $\mathcal{AH}$ is strongly one-way, and the signature scheme Π is unforgeable, the construction 7 instantiating the $flex\mathcal{RSS}$ scheme is unforgeable.*

Proof 30

Let $\mathcal{A}_{unf}$ be an algorithm winning the unforgeability game. One then uses $\mathcal{A}_{unf}$ to forge the underlying signature scheme or to calculate membership proofs. Hence, the scheme's security relies upon the security of the signature scheme and $\mathcal{AH}$. Given the game in Fig. 108 one can derive that a forgery must fall in at least one of the three cases, for at least one node in the tree:

Type 1 Forgery: The value d protected by σ_T has never been queried by $\mathcal{A}_{unf}$ to $\mathcal{O}^{Sign}$

Type 2 Forgery: The value d protected by σ_T has been queried by $\mathcal{A}_{unf}$ to $\mathcal{O}^{Sign}$, but $T^* \notin \mathsf{span}_{\vdash}(T, \mathsf{ADM})$; so the tree T^* with valid signature σ_T is not in the transitive closure of T.
This case has to be divided as well:

- *Type 2a Forgery: $T \notin \mathsf{span}_{\vdash}(T^*, \mathsf{ADM})$*
- *Type 2b Forgery: $T \in \mathsf{span}_{\vdash}(T^*, \mathsf{ADM})$*

To win $\mathcal{A}_{unf}$, the adversary must be able to construct one of the three above forgeries. This forgery can be used to break at least one of the underlying primitives.

Type 1 Forgery: *In the first case, one uses the Type 1 Forgery of $\mathcal{A}_{unf}$ to create $\mathcal{A}_{unfSig}$ which forges a signature. One constructs $\mathcal{A}_{unfSig}$ using $\mathcal{A}_{unf}$ as follows:*
(1) $\mathcal{A}_{unfSig}$ chooses a hash-function $\mathcal{AH}$ and passes prm to $\mathcal{A}_{unf}$. This is also true for pk *of the signature scheme to forge.*
(2) All queries to $\mathcal{O}^{Sign}$ from $\mathcal{A}_{unf}$ are forwarded to $\mathcal{A}_{unfSig}$'s oracle and genuinely returned to $\mathcal{A}_{unf}$.
(3) Eventually, $\mathcal{A}_{unf}$ will output a pair (T^*, σ_T^*). $\mathcal{A}_{unfSig}$ returns (T^*, σ_T^*), as a valid forgery. The concrete signature forged can easily be extracted by defining a tree-traversal algorithm looking for the signature not queried for the particular value. This is due to the fact, that the scheme allows to distribute subtrees. Hence, any node may be forged, not just the root node.

Type 2a Forgery: *In the case of 2a, one uses the Type 2a Forgery produced by $\mathcal{A}_{unf}$ to construct $\mathcal{A}_{col}$ which breaks the collision-resistance of the underlying hash-function. To do so, (1) $\mathcal{A}_{col}$ generates a key pair of a signature scheme to emulate $\mathcal{O}^{Sign}$ and chooses $\mathcal{AH}$.*
(2) It passes pk *and prm to $\mathcal{A}_{unf}$.*
(3) For every request to the signing oracle, $\mathcal{A}_{col}$ generates the signature σ using sk *and returns it to $\mathcal{A}_{unf}$.*
(4) Eventually, $\mathcal{A}_{col}$ will output (T^*, σ_T^*). Given the transcript of the simulation, $\mathcal{A}_{col}$ searches for a pair $\mathcal{MH}(n_1) = \mathcal{MH}(n_2)$ with different content resp. subtrees. If such a pair is found and $T^* \notin \mathsf{span}_{\vdash}(T_i, \mathsf{ADM}_i)$, $\mathcal{A}_{col}$ outputs exactly this pair, else it aborts. The outputted pair is a collision of the hash-function.

Type 2b Forgery: *If $\mathcal{A}_{unf}$ returns a Type 2b Forgery, one can build $\mathcal{A}_{one}$ which calculates membership proofs of the underlying accumulator. To do so, (1) $\mathcal{A}_{one}$ generates a key pair of a signature scheme to emulate $\mathcal{O}^{Sign}$ and chooses $\mathcal{AH}$.*
(2) It passes pk *and $\mathcal{AH}$ to $\mathcal{A}_{unf}$.*
(3) For every request to the signing oracle, $\mathcal{A}_{one}$ generates the signature σ using sk *and returns it to $\mathcal{A}_{unf}$.*
(4) Eventually, $\mathcal{A}_{unf}$ will output (T^*, σ_T^*). Given the transcript of the simulation, $\mathcal{A}_{one}$ searches for a pair $\mathcal{MH}(n_1) = \mathcal{MH}(n_2)$ with different content resp. subtrees. If such a pair is found and $T_i \in \mathsf{span}_{\vdash}(T^*, \mathsf{ADM}^*)$, $\mathcal{A}_{one}$ outputs $(T_i, T^*, \sigma_T, \sigma^*)$, if and only if the preimage maps to queried document. In other words, the queried tree must be in the transitive closure of the preimage. Otherwise, one just has a normal collision, which belongs to case 2a. The membership proofs of the used accumulator can trivially be extracted. This showed how to use all three forgery types to break existential unforgeability of the underlying signature scheme Π, the one-way or the collision-resistance property of $\mathcal{AH}$.
□

Theorem 31 : $flex\mathcal{RSS}$ from Construction 7 is Transparent and Private.

If $\mathcal{AH}$ always outputs uniformly distributed digests, and the digests and witnesses are therefore indistinguishable from random numbers, the construction 7 instantiating the $flex\mathcal{RSS}$ scheme is transparent and therefore also private [66].

Proof 31

This follows directly from the definitions, i.e., the uniform distribution of the digests and witnesses from random numbers. In particular, all output of $\mathcal{AH}$ is computationally indistinguishable from random. This implies that the output of $\mathcal{O}^{ModifyOrSign}$ is also computationally indistinguishable from uniform, hence hiding the secret bit b with overwhelming probability. In other words, an adversary breaking transparency is able to distinguish between random and computed digests, which has been assumed to be infeasible. Attacking the nonces is not possible, since removing a random from a uniform distribution results in a uniform distribution again. An additional note: This is the reason why $flex\mathcal{RSS}$ requires a random seed for the accumulator; otherwise, an adversary could just recalculate the digests. □

14.8.5. $flex\mathcal{RSS}$ integrity offering and visualisation

$flex\mathcal{RSS}$ is a redactable signature scheme for a tree T. It offers transparency (and therefore privacy). It is an RSS that allows for public redactions. In general this yields no detection capability for a Verifier for an occurred authorized subsequent modification. In general this also yields no option for the Signer to generate an interactive proof that a message was redacted, i.e., by providing a linking non-redacted variant of the same document. The reason is that $flex\mathcal{RSS}$ offers no non-repudiation properties on its own. However, it could inherit these from the underlying accumulating hash-function. It is out of scope of this thesis to identify if this is possible.

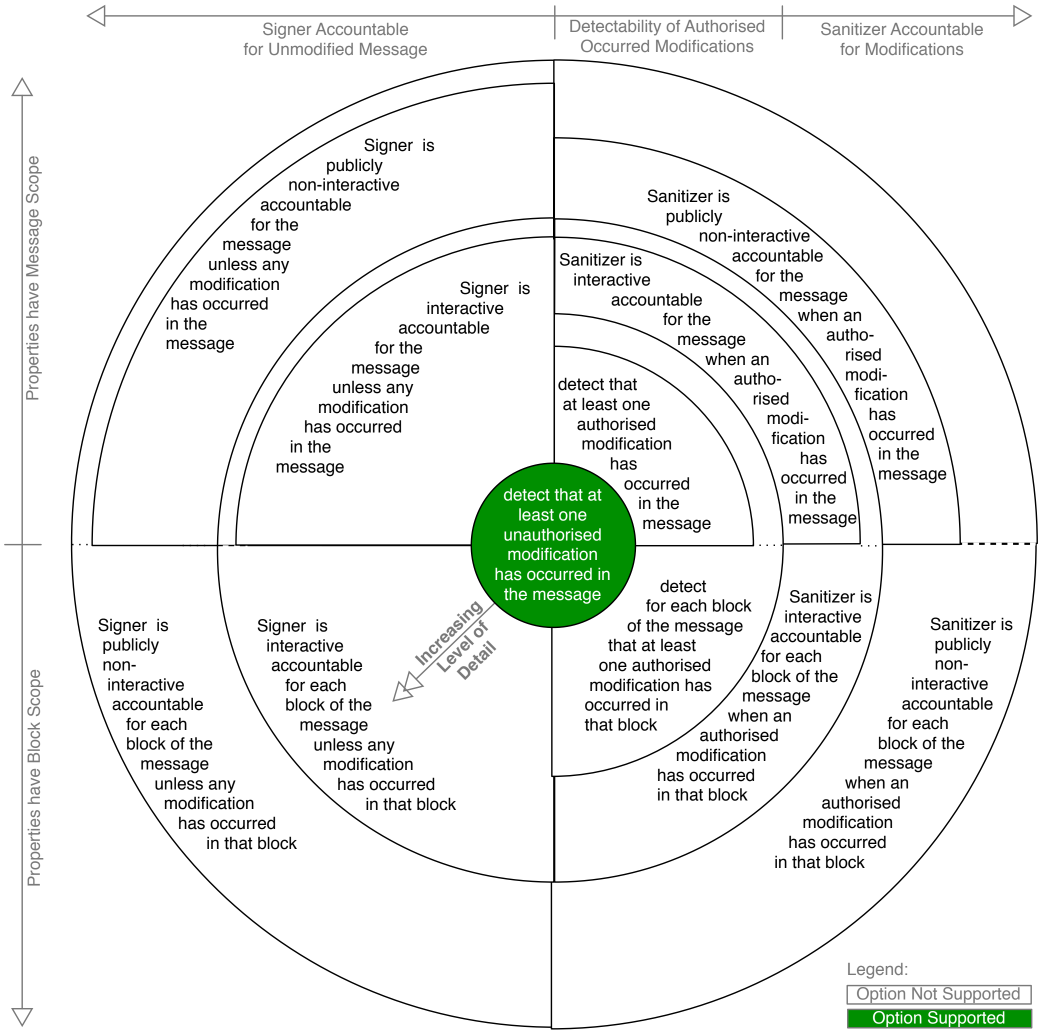

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an authorised modification has occurred in the message
Sanitizer is interactive accountable for the message when an authorised modification has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Increasing Level of Detail
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 113. $\mathit{flex}\mathcal{RSS}$ offers ACA – UCD integrity, because Verifiers can only detect unauthorized subsequent modifications and there is no accountability in $\mathit{flex}\mathcal{RSS}$; can be added by ARSS

As an ultimate solution $\mathit{flex}\mathcal{RSS}$ can be paired with an SSS that offers ACA – ≥1CD – PUB integrity following the ARSS framework (see Theorem 44 on page 422). This would make the resulting $\mathit{flex}\mathcal{RSS}$-ARSS also Sanitizer-accountable and thus would offer ACA – ≥1CD – PUB integrity.

As a safe option the worst-case behaviour based on minimal capabilities of underlying building blocks is taken for classification of $\mathit{flex}\mathcal{RSS}$. Consequently, $\mathit{flex}\mathcal{RSS}$ classifies as giving ACA – UCD integrity with the Signer staying accountable for all valid redactions. Fig. 113 visualises this.

14.9. $struct\mathcal{RSS}$ concept: RSS allowing private redaction of structure independent from content

With $struct\mathcal{RSS}$[848] this thesis shows how to maintain confidentiality of personal data or trade secrets by explicitly making the document's structure a redactable item. In structured data formats information can be stored inside the structure — and this is often the reason to use structured formats in the first place. An example for a structured data format is XML: An XML schema [496] describes the structure (and often implicitly the semantics) of a tree-structured XML document. For $struct\mathcal{RSS}$ this thesis takes linear documents as a simpler starting point.

EXAMPLE

For example: In a text file the order of subdocuments is important, e.g., the order might represent the ordering of unnumbered chapters of a book. *Liu et al.* introduced the paradigm of separating structural integrity and content integrity in [318] without considering RSS. If implicitly or explicitly information is stored inside the structure of a document but the structural integrity is not protected, because the integrity protection mechanisms is only protecting content integrity, then this obvious can lead to attacks, i.e., shuffling of list elements. The advantages of being able to redact structure individually has been given with examples in Sec. 12.5.2.

14.9.1. Shortcoming 5 addressed by $struct\mathcal{RSS}$

As said in Shortcoming 5[849], existing schemes were not able to explicitly control the redaction of structural information on its own. The goal of $struct\mathcal{RSS}$ is to allow redaction structure independent from a redaction of content to cater for pplications were such a constellation is required. $struct\mathcal{RSS}$ is transparent, meaning the fact that redaction has occurred must not be known by a verifying third party, hence $struct\mathcal{RSS}$ is also private.

14.9.2. $struct\mathcal{RSS}$ extended security model

The secure redactable signature scheme $struct\mathcal{RSS}$ achieves the following security properties for tree structured data (**bold** indicates a new property introduced in this thesis):

- Unforgeability according to Definition 172, and
- Standard Privacy following *Brzuska et al.* according to Definition 174, and
- Transparency according to Definition 175.

Additionally it offers the increased functionality of:

- Consecutive redaction control according to Subsections 14.5.1.1 and 14.9.2.1, and
- **Redaction of structural information independent from content**.

The Signer is able to redact parts of the document as well, since a RSS allows public redaction. Additionally to the standard RSS, $struct\mathcal{RSS}$ offers the added functionality of consecutive redaction control (see Sec. 14.5.1.1). Hence, all parties can sanitize and also close the document, which includes the Signer and the final recipient as well.

This section now defines the additionally required security properties of $struct\mathcal{RSS}$. Apart from disclosure control, all other properties are equal or very close to those defined already.

14.9.2.1. Consecutive redaction control: Formal security

Like $priv\mathcal{RSS}$ (see Sec. 14.5.1.1) also $flex\mathcal{RSS}$ offers to further limit the number of possible redactions. This has been termed consecutive redaction control and the definition from Sec. 14.5.1.1 was as follows:

[848] The following results, as well as the algorithm that is denoted $struct\mathcal{RSS}$, have been published in the joint work with *K. Samelin*, *A. Bilzhause* and *J. Posegga* and *H. de Meer* [426] published at ISPEC 2012 (see Appendix A publication n^{o} 13).

[849] Shortcoming 5: Limited control of the structural integrity protection (see Sec. 12.5).

Definition 218 : Consecutive Redaction / Sanitization Control

> *An* MSS *offers* ***consecutive redaction / sanitization control****, if the* $\mathsf{Sanitizer}$ *can get (or is by default configuration of the mechanism) authorized by the* Signer *to make the redaction or sanitization policy stricter. Stricter means that after adjusting* ADM *to* ADM' *the count of all admissible modifications decreases, i.e.*
>
> $$|\mathit{span}_{\vdash}(m, \mathit{ADM}')| < |\mathit{span}_{\vdash}(m, \mathit{ADM})|\,.$$
>
> *Moreover, without having access to the previous message-signature pair containing* ADM *any party seeing* $(m, \sigma, \mathsf{ADM}')$ *is not able to carry out redactions or sanitizations that have been removed (i.e. contained in* $\mathsf{ADM} \setminus \mathsf{ADM}'$*), even if initially authorized.*

In a nutshell, the security is concerned that no one should be able to redact parts of a document which are not part of ADM. Hence, an adversary might attack ADM specifically, and this is discussed in this section on the security of the disclosure control. The formal description of the attack is analogous to that on immutability for RSS (see Sec. 11.13.2) which has been transposed from its origin in SSS [64]. Note, *Miyazaki et al.* [348] merged this property with unforgeability. However, for unforgeability any message is enough to break the game; to break security of consecutive redaction control, an adversary has two possibilities: Either it is able to redact a part which is subject to redaction control or is able to alter ADM, such that the disclosure control is reversed. Therefore, the games are slightly different. Additionally, the game $\mathsf{DisclosureSecure}_{\mathcal{A}}^{\mathsf{RSS}}$ is stricter as the adversary can choose the parts of the message to be subject to disclosure control.

Definition 224 : Consecutive redaction control Security for RSS

> *A* RSS *offers* ***secure consecutive redaction control****, if for any efficient (PPT) adversary* $\mathcal{A}$ *the probability that the game* $\mathsf{DisclosureSecure}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$ *depicted in Fig. 114 returns* 1*, is negligible (as a function of* λ*).*

```
Experiment $\mathsf{DisclosureSecure}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
    $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{RSign}(\mathsf{sk},\cdot)}(\mathsf{pk})$
        where oracle RSign for input $m, \mathsf{MOD}_c$:
            $(\sigma, m) \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
            return $\mathsf{Close}(\mathsf{pk}, m, \sigma, \mathsf{MOD}_c)$
    return 1, if
        let $i = 1, 2, \ldots, q$ index the adaptive queries
        let $\mathsf{ADM}_i \leftarrow \mathsf{Modifiable}(m_i, \sigma_i)$ and
        let $\mathsf{ADM}^* \leftarrow \mathsf{Modifiable}(m^*, \sigma^*)$ and
        $\mathsf{Verify}(\mathsf{pk}, m^*, \sigma^*) = \mathtt{true}$ and
            $\exists i : m^* \in \mathsf{span}_{\vdash}(m_i) \wedge m_i \setminus m^* \not\subseteq \bigcup_{0<i\leq q} \mathsf{ADM}_i$ or
            $\exists i : \mathsf{ADM}^* \supset \mathsf{ADM}_i \wedge \forall i : \mathsf{ADM}^* \not\subseteq \bigcup_{0<i\leq q} \mathsf{ADM}_i$
```

Figure 114. Disclosure Secure Experiment for RSS that offer Consecutive Redaction Control like *structRSS*

14.9.3. *structRSS* algorithmic definition

Basically, *structRSS* has the same requirements as other RSS, i.e., unforgeability, privacy and transparency. Due to its added functionality *structRSS* needs some adjustments to the definitions, especially as *structRSS* treats structure as a redactable entity, offers consecutive redaction control. Further, *structRSS* has been defined to protect — or not to protect on the Signer's choice — the content and structural integrity of ordered lists

Definition 225 : *structRSS* Algorithms for Content and Structure Redaction

> *A* RSS *which allows separate redaction of content and structure, with consecutive redaction control, consists of five efficient (PPT) algorithms. In particular define* $\mathit{structRSS} :=$ $(\mathit{KeyGen}, \mathit{Sign}, \mathit{Verify}, \mathit{Redact}, \mathit{Close})$ *such that:*

Key Generation: *The algorithm* KeyGen *outputs the public and private key of the* Signer*, i.e.,* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, where the input parameter* λ *is the security parameter.*

Signing: *The algorithm* Sign *outputs the signature* σ *on input of the secret key* sk *and the document* m*. It outputs* $(m, \sigma) \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$*.*

Verification: *The algorithm* Verify *outputs a decision* $d \in \{\mathtt{true}, \mathtt{false}\}$ *indicating the correctness of the signature* σ*, with respect to* pk*, protecting* m*.* `true` *stands for a valid signature, while* `false` *indicates an invalid signature. In particular:* $d \leftarrow \mathsf{Verify}(\mathsf{pk}, m, \sigma)$ *.*

Redacting: *The algorithm* Redact *takes as input the document* m*, the public key* pk *of the* Signer*, the signature* σ *and description of the redaction* MOD *containing either a submessage* $m[i]$ *or a binary relation* $m[i, j]$ *that shall be redacted. Calling* Redact *sequentially allows to redact more relations and submessages. The algorithm outputs* $(m', \sigma') \leftarrow \mathsf{Redact}(\mathsf{pk}, m, \sigma, \mathsf{MOD})$*, where* $m' \leftarrow \mathsf{MOD}(m)$ *denotes the alteration of* m *with respect to* MOD*.* MOD *may contain more than one modification instruction. It is required that* ADM*, which denotes the entities of* m *admissible to be redacted, is always correctly recoverable from* (m, σ)*. The algorithm doing so will be described as:*

> **Modifiable.** *On input of a valid message-signature pair* (m, σ)*,* Modifiable *outputs* ADM*:* $\mathsf{ADM} \leftarrow \mathsf{Modifiable}(m, \sigma)$*.*
>
> $struct\mathcal{RSS}$ *defines* Modifiable *as part of* Redact *and not as a stand alone algorithm.*

Consecutive Redaction Control: *The algorithm* Close *alters* σ *on input of* m*, the public key* pk *of the* Signer*, the signature* σ *and a sanitization control description* $\mathsf{MOD_c}$ *which contains the entities subject to redaction control. The algorithm outputs* $(m, \sigma') \leftarrow \mathsf{Close}(\mathsf{pk}, m, \sigma, \mathsf{MOD_c})$*.* $\mathsf{MOD_c}$ *may contain many modifications.* Close *does not change the message* m *itself, but* $\mathsf{Modifiable}(m, \sigma') = \mathsf{Modifiable}(m, \sigma) \setminus \mathsf{MOD_c}$*. This algorithm can be called by the* Signer *and any other third party. This enables the* Signer *to close parts of* m *prior to distributing.*

14.9.4. $struct\mathcal{RSS}$ correctness

The usual RSS correctness requirements must hold (see Sec. 11.11).

14.10. $struct\mathcal{RSS}$ cryptographic instantiation

The proofs, as well as the algorithm and instantiation of $struct\mathcal{RSS}$, have been published in the joint work with *K. Samelin*, *A. Bilzhause*, *J. Posegga* and *H. de Meer* [426] published at ISPEC 2012 (see Appendix A publication n$^{\text{o}}$ 13).

14.10.1. Cryptographic preliminaries

Basically, $struct\mathcal{RSS}$ has the same requirements as *Brzuska et al.* [66], i.e., unforgeability, privacy and transparency. The definitions are slightly adjusted as $struct\mathcal{RSS}$ treats structure as a redactable entity and to make statements about linear documents instead of trees. Still it requires that the splitting of m into the blocks $m[i]$, along with their structure (order), is efficiently reconstructable from any received m. Namely, $struct\mathcal{RSS}$'s protection for linear documents assumes that the order contains statements like $m[a]$ is before $m[b]$.

14.10.1.1. Aggregate signatures and bilinear pairings

Aggregate signatures ($\mathcal{AGG}$) have been introduced by *Boneh et al.* in [58]. The basic idea is as follows:

Given ℓ signatures σ_i, $0 < i \leq \ell$, one constructs a compressed signature σ which contains all signatures σ_i. This allows verifying all given signatures σ_i by verifying σ. The scheme can be constructed as follows:

Let $\mathbb{G}_1$ be a cyclic multiplicative group with prime order q, generated by g, i.e., $\mathbb{G}_1 = \langle g \rangle$. Further, let $\mathbb{G}_T$ denote a cyclic multiplicative group with the same prime order q. Let $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ be a bilinear map such that:

1. Bilinearity: $\forall u, v \in \mathbb{G}_1 : \forall a, b \in \mathbb{Z}/q\mathbb{Z} : \hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$
2. Non-degeneracy: $\exists u, v \in \mathbb{G}_1 : \hat{e}(u, v) \neq 1$
3. Computability: There is an efficient algorithm $\mathcal{A}_{bimap}$ that calculates the mapping $\hat{e}$ for all $u, v \in \mathbb{G}_1$

14.10.1.2. *Boneh, Gentry, and Lynn* signature scheme (BGLS)

Definition 226 : The BGLS-Scheme

The $\mathcal{AGG}$ by Boneh et al. *[58] (BGLS-Scheme) with public aggregation consists of five efficient algorithms. Especially:*

$$\mathcal{AGG} = \{\textsf{A.KeyGen}, \textsf{A.Sign}, \textsf{A.Verf}, \textsf{A.Agg}, \textsf{A.AggVerf}\}$$

Key Generation: *The algorithm KeyGen outputs the public and private key of the Signer, $\textsf{sk} \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$ denote the Signer's private key and $\mathcal{H}_k : \{0,1\}^* \rightarrow \mathbb{G}_1$ an ordinary cryptographic hash-function from the family $\mathcal{H}_K$ and set $Q \leftarrow g^{\textsf{sk}}$, where g is a generator of $\mathbb{G}_1$. Set the public parameters and key $\textsf{pk} \leftarrow (g, Q, \mathbb{G}_1, \mathbb{G}_T, \mathcal{H}_k, \hat{e})$. Output $(\textsf{pk}, \textsf{sk})$.*

Signing: *The algorithm A.Sign outputs the signature σ_i on input of the secret key sk and a single document m_i. It outputs $\sigma_i \leftarrow (\mathcal{H}_k(m_i))^{\textsf{sk}}$.*

Verification: *To verify a signature σ_i, a third party has to check, if the following equation holds: $\hat{e}(\sigma_i, g) \stackrel{?}{=} \hat{e}(\mathcal{H}_k(m_i), Q)$.*

Aggregating: *To aggregate ℓ signatures σ_i, protecting m_i, into an aggregated signature σ, the aggregator computes $\sigma \leftarrow \prod_{i=1}^{\ell} \sigma_i$, denoted as A.Agg(pk, $\mathcal{S}$), where $\mathcal{S}$ is a set of signatures signed using the same public parameters. Note: This can be done by untrusted parties and without knowing the private keys.*

A.AggVerf: *To verify an aggregated signature σ, a Verifier checks whether $\hat{e}(\sigma, g) \stackrel{?}{=} \prod_{i=1}^{\ell} \hat{e}(\mathcal{H}_k(m_i), Q)$ holds, on input of σ, pk and a list of signed (sub)messages. To improve efficiency, the right side can be rewritten as $\hat{e}(\prod_{i=1}^{\ell} \mathcal{H}_k(m_i), Q)$. Note, this description uses just one public key, Q, which allows this improvement. Using just one public key also has the advantage that it is sure that just one signing key is used. The algorithm is denoted as $d \leftarrow$ A.AggVerf(pk, σ, $\{m_i\}_{0<i\leq\ell}$).*

The correctness requirements must hold; formal proofs of those can be found in [58]. Additionally, the expected security properties are required to hold, i.e., unforgeability under chosen message attacks. The proofs can also be found in [58]. It is explicitly assumed that splitting up an aggregate signature is not feasible, as shown for the BGLS-Scheme in [128]. However, this thesis requires that the adversary has access to a signing oracle. For the BGLS-Scheme, this has already been assumed in [348], but is not stated formally in [58]. Moreover, it is required that, if a third party knows a contained signature, it can build an inverse and actually remove the signature from the aggregate. This removal of σ_i from σ is denoted as $\sigma' \leftarrow \sigma \setminus \sigma_i$. For the BGLS-Scheme [58] this means: $\sigma' \leftarrow \sigma \cdot \sigma_i^{-1}$. Note, this behaviour is used by $struct\mathcal{RSS}$ to obtain a secure redaction that can be controlled by the party that can generate signatures, i.e., if known, then the signature can be removed from the aggregate, but a signature can not be added to the aggregate without knowing $\textsf{sk}_{BGLS}$.

14.10.2. $\mathit{struct}\mathcal{RSS}$ construction

The construction 8 given in the following makes use of an aggregating signature scheme $\mathcal{AGG}$ as defined earlier in Definition 226 which is based on the BGLS-Scheme given in Sec. 14.10.1.2.

Construction 8 : $\mathit{struct}\mathcal{RSS}$

Key Generation: *The key pair generation algorithm* KeyGen *outputs the key pair* $(\mathsf{sk}, \mathsf{pk})$ *using the key pair generation algorithm of the underlying aggregate signature scheme* $\mathcal{AGG}$*:* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{A.KeyGen}(1^\lambda)$.

Signing: *To sign* $m = m[1]||\dots||m[\ell]$ *perform the following steps:*

1. *Derive all "left-of" relations* $m[i,j]$ *from* m*.*
2. *Choose a nonce* τ*, i.e.,* τ *must be unique for each document* m *signed under* sk*. The tag is needed to avoid adding blocks from other documents signed with the same secret key* sk*.*

   Note, the tag τ *prohibits an adversary from mixing in blocks of other documents signed with the same secret key* sk*. Hence, a* τ *unique for each* m *'binds' all blocks to exactly one document* m^τ*.*
3. *Sign* τ*:* $\sigma_\tau \leftarrow \mathsf{A.Sign}(\mathsf{sk}, \tau)$.
   Using the BGLS-Scheme this is done as follows: $\sigma_\tau \leftarrow (\mathcal{H}_k(\tau))^{\mathsf{sk}}$.
4. *Draw* ℓ *pair-wise distinct nonces* r_i *from a uniform distribution. These are needed to prevent an adversary from aggregating a contained entity twice. They have to be drawn uniformly.*
5. *Append each* r_i *to the corresponding block* $m[i] \sqsubseteq m$*, then append* τ *and sign the resulting string:* $\sigma_i \leftarrow \mathsf{A.Sign}(\mathsf{sk}, \tau||r_i||m[i])$
   Using the BGLS-Scheme this is done as follows: $\sigma_i \leftarrow (\mathcal{H}_k(\tau||r_i||m_i))^{\mathsf{sk}}$.
6. *Sign each existing tagged "left-of" relation:*

   $$\sigma_{i,j} \leftarrow \mathsf{A.Sign}(\mathsf{sk}, \tau||r_i||r_j), \textit{for all } 0 < i < j \leq \ell, \textit{ if } m[i,j] \sqsubseteq m\,.$$

   Using the BGLS-Scheme this is done as follows:

   $$\sigma_{i,j} \leftarrow (\mathcal{H}_k(\tau||r_i||r_j))^{\mathsf{sk}}, \textit{for all } 0 < i < j \leq \ell, \textit{ if } m[i,j] \sqsubseteq m\,.$$
7. *Aggregate all generated signatures:*

   $$\sigma_c \leftarrow \mathsf{A.Agg}(\mathsf{pk}, \sigma_\tau \cup \{\sigma_i \mid m[i] \sqsubseteq m\} \cup \{\sigma_{i,j} \mid m[i,j] \sqsubseteq m\})\,.$$

   Using the BGLS-Scheme this is done as follows:

   $$\sigma_c \leftarrow \sigma_\tau \cdot \prod_{i=1}^{\ell} \sigma_i \cdot \prod_{j=2}^{\ell} \prod_{i=1}^{i<j} \sigma_{i,j}\,.$$
8. *Output* $\sigma = (\sigma_c, \tau, \{\sigma_i \mid m[i] \sqsubseteq m\}, \{\sigma_{i,j} \mid m[i,j] \sqsubseteq m\}, \{r_i \mid m[i] \sqsubseteq m \vee m[i,j] \sqsubseteq m \vee m[j,i] \sqsubseteq m\})$*.*

This algorithm already allows to sign partially ordered sets by not requiring all relations; this is necessary to maintain privacy and transparency.

Redacting Content: *The algorithm* $\mathsf{Redact}(m, k, \sigma)$ *will redact a block* $m[k]$ *from* m*, creating* $m' = m \setminus m[k]$*. To do so the following steps are performed:*

1. *Check* σ*'s validity using* Verify*. If the signature is not valid, abort and return* $\perp$*.*
2. *If* $m[k] \not\sqsubseteq m$*, abort and return* $\perp$*.*

3. *Set $m' = m \setminus m[k]$.*

   > ***Note:*** *This does not redact the block's relations, only the block's content. In order to achieve a private redaction of $m[k]$ one must invoke $\mathsf{Redact}(m, a, b, \sigma)$ for all $m[a, b]$ where $a = k$ or $b = k$; this will redact the integrity protection of the structure related to the redacted $m[k]$. The algorithm to redact structure is described below.*

4. *Remove from σ_c by deriving a new aggregated signature: $\sigma'_c = \sigma_c \setminus \sigma_k$.*
   Using the BGLS-Scheme this is computed as follows: $\sigma'_c \leftarrow \sigma_c \cdot \sigma_k^{-1}$.

5. *Destroy securely the previous old signature, the redacted $m[k]$ and the no longer required nonce r_k.*

6. *Output $\sigma' = (\sigma'_c, \tau, \{\sigma_i \mid m'[i] \sqsubseteq m'\}, \{\sigma_{i,j} \mid m'[i, j] \sqsubseteq m'\}, \{r_i \mid m'[i] \sqsubseteq m' \vee m'[i, j] \sqsubseteq m' \vee m'[j, i] \sqsubseteq m'\})$.*

Redacting Structure: *The algorithm $\mathsf{Redact}(m, a, b, \sigma)$ will redact a relation $m[a, b]$ from m. The third party has to perform the following steps to do so:*

1. *Check σ's validity using Verify. If the signature is not valid (`false`), abort and return $\perp$.*

2. *If $m[a, b] \not\sqsubseteq m$, abort and return $\perp$.*

3. *Set $m' = m \setminus m[a, b]$.*

   > *This does not redact blocks m_a nor m_b, this redacts only the relation between them. Hence, structure can be redacted individually.*

4. *Remove from σ_c by deriving a new aggregated signature: $\sigma'_c = \sigma_c \setminus \sigma_{a,b}$.*
   Using the BGLS-Scheme this is computed as follows: $\sigma'_c \leftarrow \sigma_c \cdot \sigma_{a,b}^{-1}$.

5. *Destroy securely the previous old signature and the redacted relation $m[a, b]$.*

   > *A relation $m[a, b]$ was not given its own nonce; and this operation explicitly redacts the relation only, the nonces r_a and r_b are still required as the respective blocks $m[a]$ and $m[b]$ still remain as blocks in the message m'.*

6. *Output $\sigma' = (\sigma'_c, \tau, \{\sigma_i \mid m'[i] \sqsubseteq m'\}, \{\sigma_{i,j} \mid m'[i, j] \sqsubseteq m'\}, \{r_i \mid m'[i] \sqsubseteq m' \vee m'[i, j] \sqsubseteq m' \vee m'[j, i] \sqsubseteq m'\})$.*

Verification: *The algorithm Verify performs the following steps:*

1. *Check, if all r_i are pair-wise distinct. If not, abort and return `false`.*

2. *Derive all "left-of" relations $m[i, j]$ from m.*

3. *Use $\mathsf{A.AggVerf}$ to verify the signatures over τ, over every received $m[i]$ and over all derived relations. Using the BGLS-Scheme all these checks can be done be evaluating if the following equation holds:*

$$\hat{e}(\sigma_c, g) \stackrel{?}{=} \hat{e}(\mathcal{H}_k(\tau) \cdot \prod_{i=1}^{\ell} s_i \cdot \prod_{j=2}^{\ell} \prod_{i=1}^{i<j} t_{i,j}, Q)$$

   where

$$s_i = \begin{cases} \mathcal{H}_k(\tau || r_i || m[i]) & \text{if } m[i] \sqsubseteq m \\ 1 & \text{otherwise} \end{cases} \text{ and } t_{i,j} = \begin{cases} \mathcal{H}_k(\tau || r_i || r_j) & \text{if } m[i, j] \sqsubseteq m \\ 1 & \text{otherwise} \end{cases} .$$

   If all validations pass, then return `true`, otherwise `false` resp. $\perp$ on error.

If this test is passed then the ordering and the content have both been verified explicitly *and the received document is valid. Being given a block or a block's relation which is not part of the received document, does not impact on transparency, since the document could have been signed like this. Still, it is assumed that the* Verifier *knows which information is part of the signature and which not; this provides the basis for a useable verification procedure, because if found to be neither part of* m, s_i *nor* $t_{i,j}$ *the procedure defaults back to* 1, *the neutral element in the multiplicative group. Moreover, no information from the signature is leaked, as required by the security model. Thus, the instantiation built upon the BGLS-Scheme gives a very compact representation of the verification algorithm.*

Redaction Control: *The algorithm* Close *prohibits the possibility of further redaction:*

1. *Check the validity of* σ *using* Verify. *If the signature is not valid* (`false`), *then abort and return* $\perp$.
2. *If a block* $m[k]$ *is subject to redaction control, do not distribute* σ_k *anymore:* $\sigma' = (\sigma_c, \tau, \{\sigma_i \mid m[i] \sqsubseteq m \wedge \sigma_i \neq \sigma_k\}, \{\sigma_{i,j} \mid m[i,j] \sqsubseteq m\}, \{r_i \mid m[i] \sqsubseteq m \vee m[i,j] \sqsubseteq m \vee m[j,i] \sqsubseteq m\})$.
3. *If a block relation* $m[i,j]$ *is subject to redaction control, then do not distribute* $\sigma_{i,j}$ *anymore:* $\sigma' = (\sigma_c, \tau, \{\sigma_i \mid m[i] \sqsubseteq m\}, \{\sigma_{i,j} \mid m[i,j] \sqsubseteq m\}, \{r_i \mid (m[i] \sqsubseteq m \vee m[i,j] \sqsubseteq m \vee m[j,i] \sqsubseteq m) \wedge \sigma_{i,j} \neq \sigma_{k,l}\})$.

The algorithms Redact and Close do not require any private keys. They only allow to 'remove' or to 'close' just a single block or one relation. This is done for brevity; sequentially running the given algorithms re-establishes the required and intuitive behaviour, i.e., removing a block along with its relations. The reason why construction 8 adds σ_τ to the aggregate is as follows: If at least one block or one relation is closed, an adversary must be able to extract σ_τ, which is later proven to be infeasible. Thus, the verification algorithm will not accept the signature, which re-establishes the required correctness requirement. A third party having all signatures but σ_τ can calculate it by redacting all relations and all blocks. However, this does not introduce any security problems, since the third party could give away all signatures anyway.

Above construction 8 is generic and to clarify the implementation using the BGLS-Scheme was woven in. The proof given in the Section 14.10.4 is done on the basis of the BGLS-Scheme as well.

14.10.3. $struct\mathcal{RSS}$ runtime and complexity

The above construction for $struct\mathcal{RSS}$ has been implemented as part of the joint work [426] with *K. Samelin*, *A. Bilzhause*, *J. Posegga* and *H. de Meer* published at ISPEC 2012 (see Appendix A publication n^{o} 13).

The results obtained demonstrate the usability despite its runtime complexity of $\mathcal{O}(n^2)$ and the fact that it is based on pairings. For the implementation of pairings the proof of concept implementation used the library developed by the *National University of Maynooth*[850] [366]. All tests were run on a laptop with an *Intel* T8300 Dual Core @2.40 GHz and 4 GiB of RAM. The software was *Ubuntu* Version 10.04 LTS (64 Bit) and Java version `1.6.0_26-b03`. A single thread was used to calculate the signatures; hence an improvement would be to parallelise signature calculations, since all but the aggregation step are independent. The measurement were repeated and the median of 10 runs was calculated. Further, the measurements evaluated three sizes of curves, i.e., 128, 256 and 384 Bit. Tab. 24 shows the results for messages with 10, 25, 50 and 100 blocks.

As shown, for high security parameter sizes and high block counts, $struct\mathcal{RSS}$ becomes considerably slower than a standard hash-function, like SHA-512. For comparison, a SHA-512 on a document with 10 block takes 4ms and for 100 it takes 40ms. So, the implementation of $struct\mathcal{RSS}$ is at best $1,587$ times slower than SHA-512 (10 block signed using a 128bit curve). In comparison to other primitives based on pairings used in SSS the $struct\mathcal{RSS}$ scheme can compete: A chameleon hash-function like *Zhang* et. al's [511] (128bit) takes 930ms to generate a single hash according to measurements calculated for

[850] http://www.nuim.ie/ [last accessed: Jan. 2018].

	Generation of σ in *ms*			
ℓ Curve	10	25	50	100
128 Bit	6,350	28,675	158,557	615,546
256 Bit	39,313	170,405	667,321	2,660,354
384 Bit	95,555	435,902	1,740,444	6,837,645

	Verification of σ in *ms*			
ℓ Curve	10	25	50	100
128 Bit	3,675	16,638	89,233	338,156
256 Bit	20,323	92,828	345,360	1,401,178
384 Bit	49,203	229,935	896,825	3,580,709

	Redaction in *ms*			
ℓ Curve	10	25	50	100
128 Bit	3	10	22	32
256 Bit	9	20	44	83
384 Bit	15	37	71	153

Table 24. $\mathit{struct}\mathcal{RSS}$ instantiation's median runtime in ms

the jointly published technical report [392]. The $\mathit{struct}\mathcal{RSS}$ scheme has a growth of $\mathcal{O}(n^2)$. However, all other *provably* secure[851] and transparent schemes, i.e., [66] and [117], have the same complexity and therefore just differ by a constant factor. Hence, faster aggregate signatures would directly lead to a faster scheme. Note that for large security parameters, and a large submessage count, the scheme becomes very slow.

14.10.4. $\mathit{struct}\mathcal{RSS}$ security proof

The $\mathit{struct}\mathcal{RSS}$ from Construction 8 is secure, i.e. correct, unforgeable, private, transparent, and offers secure consecutive redaction control security. Each property is proven on its own.

Theorem 32 : $\mathit{struct}\mathcal{RSS}$ from Construction 8 is Correct.

> *The Construction 8 instantiating $\mathit{struct}\mathcal{RSS}$ is correct.*

Proof 32

> *Trivially follows from the definitions and the algorithms, i.e., every information claimed to be valuable is explicitly signed and must explicitly be verified. Signing the "left-of" relationship is enough to protect the ordering due to the transitive behaviour. See [117] and [66] for additional information. Note, the unique nonces imply that copy attacks, i.e., just aggregating a specific signature σ_i again are prohibited. The nonces also circumvent the problem of not being able to sign a document where a document is contained twice from the beginning [58].* □

Theorem 33 : $\mathit{struct}\mathcal{RSS}$ from Construction 8 is Private.

> *The construction 8 instantiating $\mathit{struct}\mathcal{RSS}$ is private in the information-theoretical sense.*

Proof 33

> *The instantiation of the $\mathit{struct}\mathcal{RSS}$ scheme is private in the information-theoretical sense. In particular, the redacted blocks are completely removed from the signature and the message. Hence, the secret bit b is perfectly hidden. The signing algorithm requires that always fresh r_i are drawn* uniformly, *while removing a random number from a uniformly distributed list leads to a uniformly distributed list again. Hence, even an unbounded adversary is not able to guess the bit better than at random. The adversary would be able to distinguish between two uniform distributions. The other way around is similar; if the redacted*

[851] As defined in Sec. 2.3.1.

message would have been signed directly, while the corresponding r_i are not changed, the output is the same, prohibiting even unbounded adversary's from guessing any better than random. This implies perfect privacy, meaning private in an information-theoretical sense. □

Theorem 34 : *struct$\mathcal{RSS}$* from Construction 8 is Transparent.

The construction 8 instantiating struct$\mathcal{RSS}$ is transparent in the information-theoretical sense (perfect transparency).

Proof 34

The instantiation of the struct$\mathcal{RSS}$ scheme is also transparent in the information-theoretical sense. In other words, the secret bit b is perfectly hidden. The signing algorithm requires that always fresh r_i are drawn uniformly*, while removing a random number from a uniformly distributed list leads to a uniformly distributed list again. Hence, even an unbounded adversary is not able to guess the bit better than at random. Otherwise, the adversary would be able to distinguish between two uniform distributions, which is obviously impossible. Again, the other way around is similar: If the redacted message had been signed directly, the distributions were still uniform and it is impossible for any adversary to guess b better than at random.* □

Theorem 35 : *struct$\mathcal{RSS}$* from Construction 8 is Unforgeable.

The construction 8 instantiating struct$\mathcal{RSS}$ is unforgeable.

Proof 35

Note, struct$\mathcal{RSS}$ required that the tags τ_m are chosen uniquely for each message, while sk *is fixed. The r_i are drawn uniformly as well. Hence, the following analysis will omit unlikely collisions of those and trivial mix-and-match-attacks. Knowing this, one can construct an adversary $\mathcal{B}$ with breaks the unforgeability of the BGLS-Scheme, if an adversary $\mathcal{A}$ with a non-negligible advantage ϵ exists, winning the unforgeability game. To do so, $\mathcal{B}$ uses $\mathcal{A}$ as a black box. For every signature query $\mathcal{A}$ requests, $\mathcal{B}$ forwards the queries to its signing oracle $\mathcal{O}^{Sign}$ and genuinely returns the answers to $\mathcal{A}$. Eventually, $\mathcal{A}$ will output a pair (m^*, σ^*). Given the transcript of the simulation, $\mathcal{B}$ checks, if (m^*, σ^*) is a trivial "forgery", i.e., a result of an allowed redaction. If so, $\mathcal{B}$ aborts the simulation. If, at some time, $\mathcal{B}$ does not need to restart, $\mathcal{B}$ outputs the tuple (m^*, σ^*) as its forgery attempt. Note, if $\exists i : \sigma^* \neq \sigma_i \wedge m_i = m^*$, then the pair (m^*, σ^*) does not win the unforgeability game (see Definition 172 on page 287) and is therefore not a valid forgery attempt. This ends the simulation.*

One has to distinguish between two cases:

(1) If $\exists i : \sigma^* = \sigma_i \wedge m_i \neq m^*$, $\mathcal{B}$ has found collision of the underlying random oracle or must have forged at least two messages. One can extract the colliding aggregates and output them as a valid forgery of the BGLS-Scheme itself. In both cases, m^* has never been queried.

(2) If $\neg\exists i : \sigma^* = \sigma_i \vee m_i = m^*$. Then one obtains a valid forgery of a message m never queried. This breaks the unforgeability of the BGLS-Scheme.

The proof how $\mathcal{B}$ will be used to break the "Diffie-Hellman-Problem" is discussed in [58] and [348]. The authors, Boneh et al. and Miyazaki et al. show that $\mathcal{B}$ always outputs a valid forgery, if $\mathcal{A}$ is successful. Hence, also with the probability of $\mathcal{A}$, which is ϵ. □

Theorem 36 : *struct$\mathcal{RSS}$* from Construction 8 offers secure consecutive redaction control.

The construction 8 instantiating struct$\mathcal{RSS}$ offers secure consecutive redaction control.

Proof 36

Note, the following proofs are given under the assumption, that the k-element-aggregate-extraction-assumption (k-EAEA) [58] from the BGLS yields, even if the adversary has access to a signing oracle. Let $\mathcal{A}$ be an algorithm winning the game for consecutive redaction control security (disclosure secure game from Definition 224 on page 400). The adversary will use $\mathcal{A}$ to break k-EAEA and therefore also the "Diffie-Hellman Assumption" [128] resp. the unforgeability of the underlying BGLS-Scheme. To do so, let $\mathcal{B}$ use $\mathcal{A}$ as a black-box again. For every query of $\mathcal{A}$ to the oracle $\mathcal{O}^{RSign}$, $\mathcal{B}$ forwards the queries to its oracle $\mathcal{O}^{A.AggSign}$, where $\mathcal{O}^{A.AggSign}$ signs and aggregates messages in one step. For all messages $m_c = m_i \setminus \mathsf{MOD}_{c,i}$, $\mathcal{B}$ calls $\mathcal{O}^{Sign}$ and simulates $(m_i', \sigma_i') \leftarrow \mathsf{Close}(pk, m_i, \sigma_i, \mathsf{MOD}_{c,i})$. Afterwards, it forwards (m_i', σ_i') genuinely to $\mathcal{A}$. Eventually, $\mathcal{A}$ outputs its forgery attempt (m^*, σ^*). If (m^*, σ^*) is non-trivial and actually winning the disclosure secure game, $\mathcal{B}$ outputs (m^*, σ^*), otherwise $\mathcal{B}$ aborts. This ends the description of the simulation. There are two cases:

Case 1: *$\mathcal{A}$ could reconstruct parts of ADM_i*

Case 2: *$\mathcal{A}$ could redact parts of m_i, which were subject to disclosure control*

The first case: Trivial; $\mathcal{A}$ must be able to extract signatures from the aggregate; The algorithm given in [128] can use $\mathcal{B}$ to break the DH-Assumption. The second case: Either $\mathcal{A}$ forged the signature or extracts sub-signatures as well. As before, an algorithm able to forge signatures can be used to solve the DH-Problem [58, 348]. To extract the sub-signatures in the non-forgery case, one reverse-calculates the signatures. In particular, one only needs to calculate $\sigma \setminus \sigma^*$ and output the result. The result was an aggregated signature, and therefore the algorithm given in [128] can use $\mathcal{B}$ to break DH. □

14.10.5. *struct$\mathcal{RSS}$* integrity offer and visualisation

struct$\mathcal{RSS}$ is a redactable signature scheme for an ordered lists. It offers transparency (and therefore privacy) even in the information theoretical sense (see Theorem 34 and the related proof). It is an RSS that allows for public redactions and also public consecutive redaction control. In sum this yields no detection capability for a Verifier for an occurred authorized subsequent modification.

As an ultimate solution *flex$\mathcal{RSS}$* can be paired with an SSS that offers ACA – ≥1CD – PUB integrity following the ARSS framework (see Theorem 44 on page 422). This would make the resulting *flex$\mathcal{RSS}$*-ARSS also Sanitizer-accountable and thus would offer ACA – ≥1CD – PUB integrity.

struct$\mathcal{RSS}$ on its own and in its basic construction classifies as giving ACA – UCD integrity as the Signer stays accountable for all valid redactions. Fig. 115 visualises this.

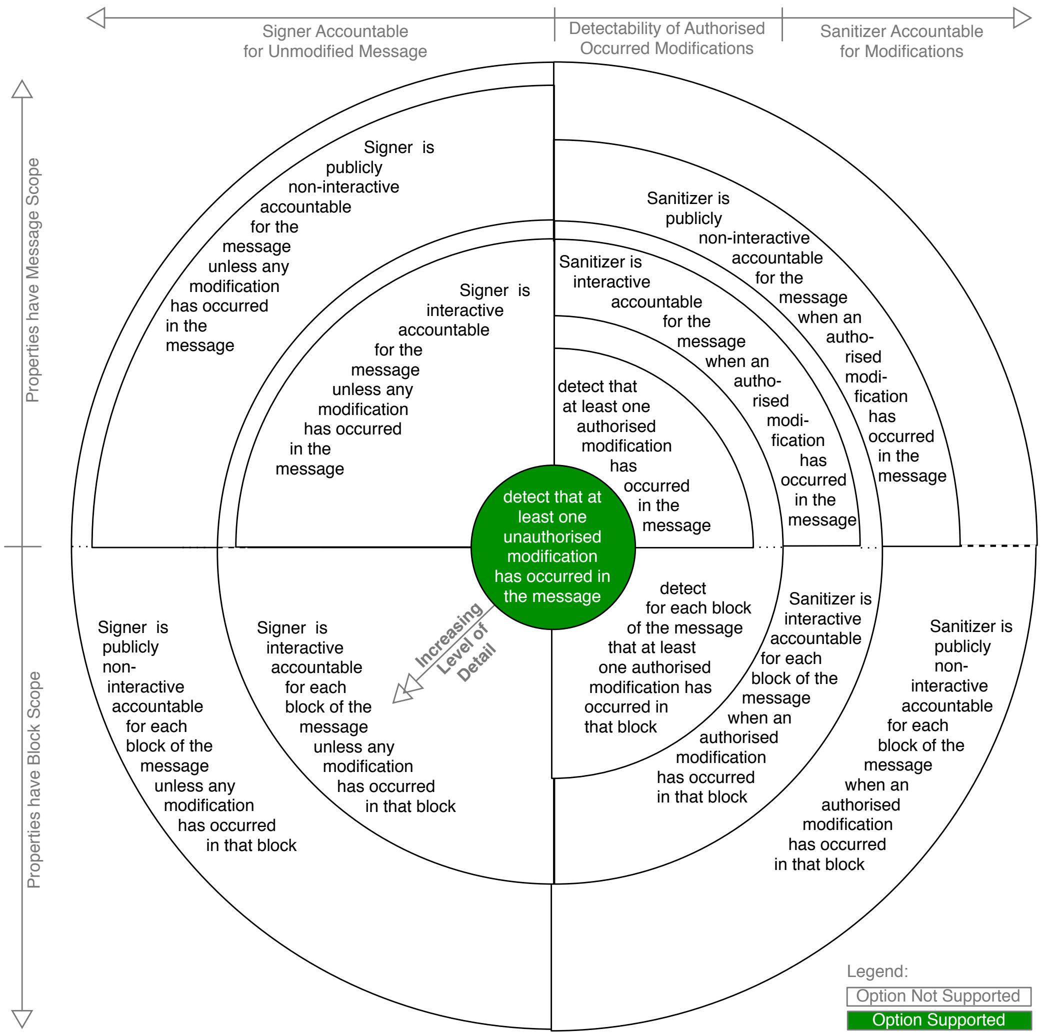

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Increasing Level of Detail
Legend:
Option Not Supported
🟩 Option Supported

Figure 115. *struct*$\mathcal{RSS}$ offers ACA – UCD integrity, because Verifiers can only detect unauthorized subsequent modifications and there is no accountability in *struct*$\mathcal{RSS}$; can be added by ARSS

14.11. *merge*$\mathcal{RSS}$ Concept: Allow merging several redacted documents into one, if and only if they stem from the same signed source

The *merge*$\mathcal{RSS}$ scheme described in this section was published as joint work with *K. Samelin*, *J. Posegga* and *H. de Meer* at the ACNS conference in 2014 [387] (see Appendix A publication n° 20). Former to that, an earlier version with extended results was jointly published with *K. Samelin*, *J. Posegga* and *H. de Meer* as a technical report [393] (see Appendix A publication n° 25) in 2012. The scheme was additionally implemented on a smart card (see Sec. 14.15).

14.11.1. *merge*$\mathcal{RSS}$ extended security properties

A valid redaction generates a new variant[852] of a document.

The redactable signature scheme *merge*$\mathcal{RSS}$ achieves the following security properties (**bold** indicates a new security property introduced in this thesis):

- Unforgeability according to Definition 172, and

[852] Using the term ‘variant’ instead of version to avoid thinking of version control systems that allow to revert subsequent modifications. RSS have the purpose to not allow to revert the redaction, i.e., property of privacy.

- Standard Privacy following *Brzuska et al.* according to Definition 174, and additionally
  - **merge privacy** according to Definition 209, and
  - **update privacy** according to Definition 211,
- Transparency according to Definition 175, and additionally
  - **merge transparency** according to Definition 210, and
  - **update transparency** according to Definition 212.
- Linkability according to Sec. 14.4.

It does offer the new explicit functionality of merging and updating and with it added flexibility.

14.11.2. *merge$\mathcal{RSS}$* algorithmic description

merge$\mathcal{RSS}$ instantiates the algorithms given for the MRSS framework (see Sec. 14.2.2).

14.12. *merge$\mathcal{RSS}$* cryptographic instantiation

The *merge$\mathcal{RSS}$* scheme's cryptographic instantiation and security proofs were published [387] at the Applied Cryptography and Network Security (ACNS) conference in 2014 as joint work with *H. de Meer*, *J. Posegga* and *K. Samelin* (see Appendix A publication n$^{\text{o}}$ 20). Former to that, an earlier version with extended results was jointly published with the same co-authors as a technical report [393] (see Appendix A publication n$^{\text{o}}$ 25) in 2012. The scheme was also implemented on a commercially available smart card (see Sec. 14.15), which required a new more smart-card-suitable instantiation of an accumulator (see Sec. 14.15.2.2).

14.12.1. Cryptographic preliminaries

merge$\mathcal{RSS}$ scheme can be instantiated from *strongly collision-resistant trapdoor accumulators* (like the one presented in Sec. 14.12.1.2) and *standard digital signatures* (like RSASSA-PSS from PKCS-v2.2 [422]).

14.12.1.1. Trapdoor accumulators

Cryptographic accumulators have been introduced by *Benaloh* and *de Mare* [43]. They hash a potentially very large set $\mathcal{S}$ into a short single value a, called the accumulator. For each element accumulated, a witness is generated, which vouches for the accumulation. A trapdoor-accumulator allows generating proofs for new elements not contained by use of a trapdoor. The following construction is based upon such an accumulator. Using such an accumulator allows to achieve the mergeability of *merge$\mathcal{RSS}$* directly, as one can add and remove witnesses and the corresponding elements freely. The property of non-membership witnesses [312], or non-deniability [316] is not required for *merge$\mathcal{RSS}$* to work. The possibility of dynamically updating an accumulator [104] is also not necessary for *merge$\mathcal{RSS}$*'s goals.

The following definition is derived from *Barić and Pfitzmann* [24].

Definition 227 : Trapdoor Cryptographic Accumulators

> *A cryptographic trapdoor accumulator* ACC *consists of four efficient (PPT) algorithms. In particular,* $\mathsf{ACC} := (\mathsf{Gen}, \mathsf{Dig}, \mathsf{Proof}, \mathsf{Verf})$ *such that:*
>
> ***Key Generation:*** *The algorithm* Gen *is the key generator. On input of the security parameter* λ*, it outputs the key pair* $(\mathsf{sk_{ACC}}, \mathsf{pk_{ACC}}) \leftarrow \mathsf{Gen}(1^{\lambda})$.
>
> ***Digest:*** *The algorithm* Dig *takes as input the set* $\mathcal{S}$ *to accumulate, the public parameters* $\mathsf{pk_{ACC}}$*. It outputs an accumulator value* $a \leftarrow \mathsf{Dig}(1^{\lambda}, \mathsf{pk_{ACC}}, \mathcal{S})$.
>
> ***Witness Generation:*** *The deterministic algorithm* Proof *takes as input the secret key* $\mathsf{sk_{ACC}}$*, the accumulator* a*, and a value* v *and returns a witness* p *for* v*. Hence, it outputs* $p \leftarrow \mathsf{Proof}(1^{\lambda}, \mathsf{sk_{ACC}}, a, v)$.

```
**Experiment** $\mathsf{Strong} - \mathsf{Coll.} - \mathsf{Res.}_{\mathcal{A}}^{\mathsf{ACC}}(\lambda)$
  $(\mathsf{sk}_{\mathsf{ACC}}, \mathsf{pk}_{\mathsf{ACC}}) \leftarrow \mathsf{Gen}(1^\lambda)$
  $(S^*, st) \leftarrow \mathcal{A}_1(1^\lambda, \mathsf{pk}_{\mathsf{ACC}})$ //$st$ denotes $\mathcal{A}$'s state
  $a \leftarrow \mathsf{Dig}(1^\lambda, \mathsf{pk}_{\mathsf{ACC}}, S^*)$
  $(v^*, p^*) \leftarrow \mathcal{A}_2^{\mathsf{Proof}(1^\lambda, \mathsf{sk}_{\mathsf{ACC}}, a, \cdot)}(st, a)$
  return 1, if
    $\mathsf{Verf}(1^\lambda, \mathsf{pk}_{\mathsf{ACC}}, a, v^*, p^*) = 1$,
    and $v^*$ has not been queried to $\mathsf{Proof}$
```

Figure 116. Strong Collision-Resistance

Verification: *The verification algorithm* Verf *takes as input the public key* $\mathsf{pk}_{\mathsf{ACC}}$*, an accumulator* a*, a witness* p*, and a value* v *and outputs a bit* $d \in \{0, 1\}$*, indicating whether* p *is a valid* $(d = 1)$ *witness for* v *with respect to* a *and* $\mathsf{pk}_{\mathsf{ACC}}$*. Hence, it outputs* $d \leftarrow \mathsf{Verf}(1^\lambda, \mathsf{pk}_{\mathsf{ACC}}, a, v, p)$.

The usual correctness properties for accumulators, as formally defined by Barić and Pfitzmann [24], are required to hold.

Strong Collision-Resistance. An adversary should not be able find a valid witness/element pair (p^*, v^*) for a given accumulator a, even if it is allowed to adaptively query for elements not contained in the original set accumulated and to choose the original set to be accumulated. A family of trapdoor accumulators is called *strongly collision-resistant*, if the probability that the experiment depicted in Fig. 116 returns 1, is negligible. Note, this definition is very similar to the standard unforgeability of signature schemes. The naming of this property is due to historical reasons [24].

14.12.1.2. Trapdoor accumulator instantiation (ACC)

The following trapdoor accumulator instantiation ACC is built using the ideas given by *Barić and Pfitzmann* [24], but makes use of the trapdoor $\varphi(n)$.

Construction 9 : Trapdoor-Accumulator ACC

As a prerequisite the following requires a division-intractable hash-function $\mathcal{H} : \{0, 1\}^* \to \{0, 1\}^\lambda$ *mapping to odd numbers. A formal definition of such a function is given in [202]. Let* $\mathsf{ACC} := (\mathsf{Gen}, \mathsf{Dig}, \mathsf{Proof}, \mathsf{Verf})$ *such that:*

Key Generation: *Generate* $n = pq$*, where* p *and* q *are distinct safe primes of length* λ.[853] *Return* $(\varphi(n), (n, \mathcal{H}))$*, where* $\varphi(pq) := (p-1) \cdot (q-1)$.

Digest: *To improve efficiency* Dig *uses the built-in trapdoor. A new digest can therefore be drawn at random. Return* $a \in_R \mathbb{Z}_n^\times$.

Witness Generation: *To generate a witness* p_i *for an element* v_i*, set* $v_i' \leftarrow \mathcal{H}(v_i)$*. Return* $p_i \leftarrow a^{v_i'^{-1} \pmod{\varphi(n)}} \bmod n$.

Verification: *To check the correctness of a proof* p *with respect to an accumulator* a*, the public key* $\mathsf{pk}_{\mathsf{ACC}}$*, and a value* v*, the* Verf *algorithm will return* 1*, if* $a \overset{?}{=} p^{\mathcal{H}(v)} \pmod{n}$*, and* 0 *otherwise.*

Note that this construction is related to GHR-signatures [202]. Due to the built-in trapdoor, it does not require any auxiliary information as proposed in [24]. The use of safe primes allows to almost always find a root for odd numbers. If one is not able to do so, one can trivially factor n. The proofs that a trapdoor-accumulator constructed as such is strongly collision-resistant is as follows:[854]

Theorem 37 : The Accumulator is Strongly Collision-Resistant.

The trapdoor-accumulator ACC *from Construction 9 is strongly collision-resistant.*

[853] A prime p is safe, if $p = 2p' + 1$, where p' is also prime.

[854] The proof can be found in the appendix of the joint publication [387] at the Applied Cryptography and Network Security (ACNS) conference in 2014 (see Appendix A publication n° 20).

Proof 37

Let $\mathcal{A}$ be an adversary breaking the strong-collision-resistance of the underlying accumulator. One can then turn $\mathcal{A}$ into an adversary $\mathcal{B}$ which breaks the unforgeability of the GHR-signature [202] in the following way:

1. *$\mathcal{B}$ receives the modulus n, the hash-function $\mathcal{H}$, and the value s. All is provided by the GHR-challenger*
2. *$\mathcal{B}$ sends $\mathsf{pk} = (n, \mathcal{H})$ to $\mathcal{A}$. Then, $\mathcal{B}$ waits for $\mathcal{S}$ from $\mathcal{A}$*
3. *$\mathcal{B}$ sends s to $\mathcal{A}$. Note, one obtains a perfect simulation here, even as $\mathcal{S}$ is ignored, as the GHR-signature scheme draws s in the exact same way as done for this accumulator*
4. *For each Proof-oracle query v_i, $\mathcal{B}$ asks its signing oracle provided, which returns a signature σ_i. Send σ_i as the witness p_i back to $\mathcal{A}$*
5. *Eventually, $\mathcal{A}$ terminates with an attempted forgery (v^*, p^*)*
6. *$\mathcal{B}$ returns (v^*, p^*) as its own forgery attempt* □

Now let $y = v^*$, and $p = \sigma^*$. As $s = p^{\mathcal{H}(y)} \pmod{n}$, and one has embedded the required challenges accordingly, $\mathcal{B}$ breaks the GHR-signature with the same probability as $\mathcal{A}$ breaks the strong collision-resistance of the proposed trapdoor-accumulator. *Gennaro et al.* [202] shows how to break the strong-RSA-assumption with the given forgery.

Note, that an adversary can simulate the Proof-oracle itself for the elements used for Dig. It calculates $a = x^{\prod_{v_i \in S} \mathcal{H}(v_i)} \mod n$ for a random $x \in_R \mathbb{Z}_n^\times$ and for each proof p_i, it lets $p_i = x^{\prod_{v_j \in S, i \neq j} \mathcal{H}(v_j)} \mod n$. For new elements, this technique does not work. Note, a is drawn at random for efficiency, as stated previously this requires to use the built-in trapdoor. This has no impact on the security: a will be distributed exactly in the same way if the slower method would be used.

14.12.2. *merge$\mathcal{RSS}$* construction

The idea behind the construction of *merge$\mathcal{RSS}$* is as follows:

1. Fix the accumulator a for *all* signatures. Additionally, each element is tagged with a unique string τ to tackle mix-and-match attacks. Hence, all derived subset-signature pairs are linkable by the tag τ. τ is also accumulated to avoid trivial "empty-set"-attacks.
2. Redactions remove v_i and its corresponding witness p_i (also called proofs). The redactions are private, as without knowledge of the witness p_i nobody can verify if v_i is "in" the accumulator a.
3. Mergeability is achieved, as supplying an element-witness pair allows a third party to add it back into the signature.
4. Unforgeability is inherited from the use of a strong collision-resistant accumulator ACC.
5. Dynamic updates are possible due to a trapdoor in ACC, only known to the Signer.
6. Privacy directly follows from definitions, i.e., the number of witnesses is fixed, while the witnesses themselves are deterministically generated, without taking already generated witnesses into account.
7. Due to mergeability the scheme is linkable, which together with a history of all signatures kept by the Signer allows to simulate Proof and Judge algorithms formerly known only in the realm of SSS.

Note, a very straight-forward construction exists: Sign each element $v_i \in \mathcal{S}$ and give out all the signatures. *merge$\mathcal{RSS}$* is advantageous compared to that as the underlying accumulator can be exchanged to inherit or derive new properties, e.g., prohibiting updates using a trapdoor-free accumulator [316].

The instantiation of $merge\mathcal{RSS}$ proves that using accumulators is sufficient, opposing the results given by *Kundu and Bertino* [299]. They stated that the "[...] use of accumulators to compute the Merkle hash signature [...] cannot prevent the leakage of extraneous information about a node and the structural relationships with its descendants or ancestors." *Kundu and Bertino* [299].

Construction 10 : $merge\mathcal{RSS}$: An Updatable and Mergeable RSS (MRSS)

Let $merge\mathcal{RSS} := (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Redact}, \mathsf{Update}, \mathsf{Merge}, \mathsf{Link})$ such that:

Key Generation: *The algorithm KeyGen generates the key pair in the following way:*

1. *Generate key pair required for ACC, i.e., run $(\mathsf{sk_{ACC}}, \mathsf{pk_{ACC}}) \leftarrow \mathsf{Gen}(1^\lambda)$;*
2. *Call $a \leftarrow \mathsf{Dig}(\mathsf{pk_{ACC}}, \emptyset)$;*
3. *Output $(\mathsf{sk_{ACC}}, (\mathsf{pk_{ACC}}, a))$.*

Signing: *To sign a set $\mathcal{S}$, perform the following steps:*

1. *Draw a tag $\tau \in_R \{0,1\}^\lambda$;*
2. *Let $p_\tau \leftarrow \mathsf{Proof}(\mathsf{sk_{ACC}}, a, \tau)$;*
3. *Output $(\mathcal{S}, \sigma, \tau)$, where $\sigma = (p_\tau, \{(v_i, p_i) \mid v_i \in \mathcal{S} \wedge p_i \leftarrow \mathsf{Proof}(\mathsf{sk_{ACC}}, a, v_i||\tau)\})$;*

Verification: *To verify signature $\sigma = (p_\tau, \{(v_1, p_1), \ldots, (v_k, p_k)\})$ with tag τ, perform:*

1. *For all $v_i \in \mathcal{S}$ check that $\mathsf{Verf}(\mathsf{pk_{ACC}}, a, v_i||\tau, p_i) = \texttt{true}$;*
2. *Check that $\mathsf{Verf}(\mathsf{pk_{ACC}}, a, \tau, p_\tau) = \texttt{true}$;*
3. *If Verf succeeded for all elements, output `true`, otherwise `false`.*

Redacting: *To redact a subset $\mathcal{R}$ from a valid signed set $(\mathcal{S}, \sigma)$ with tag τ, with $\mathcal{R} \subseteq \mathcal{S}$, the algorithm performs the following steps:*

1. *Check the validity of σ using Verify. If σ is not valid (`false`), return $\perp$;*
2. *Output $(\mathcal{S}', \sigma', \tau)$, where $\sigma' = (p_\tau, \{(v_i, p_i) \mid v_i \in \mathcal{S} \setminus \mathcal{R}\})$.*

Updating: *To update a valid signed set $(\mathcal{S}, \sigma)$ with tag τ by adding $\mathcal{U}$ and knowing $\mathsf{sk_{ACC}}$, the algorithm performs the following steps:*

1. *Verify σ with respect to τ using Verify. If σ is not valid, return $\perp$;*
2. *Output $(\mathcal{S} \cup \mathcal{U}, \sigma', \tau)$, where $\sigma' = (p_\tau, \{(v_i, p_i) \mid v_i \in \mathcal{S}\} \cup \{(v_k, p_k) \mid v_k \in \mathcal{U}, p_k \leftarrow \mathsf{Proof}(\mathsf{sk_{ACC}}, a, v_k||\tau)\})$.*

Merging: *To merge two valid set-signature pairs $(\mathcal{S}, \sigma_\mathcal{S})$ and $(\mathcal{T}, \sigma_\mathcal{T})$ with an equal tag τ, the algorithm performs the following steps:*

1. *Verify $\sigma_\mathcal{S}$ and $\sigma_\mathcal{T}$ with respect to τ using Verify; If they do not verify (`false`), return $\perp$;*
2. *Check, that both have the same tag τ;*
3. *Output $(\mathcal{S} \cup \mathcal{T}, \sigma_\mathcal{U}, \tau)$, where $\sigma_\mathcal{U} = (p_\tau, \{(v_i, p_i) \mid v_i \in \mathcal{S} \cup \mathcal{T}\})$, where p_i is taken from the corresponding signature.*

Link: *To identify if two sets $(\mathcal{S}, \sigma_\mathcal{S})$ and $(\mathcal{T}, \sigma_\mathcal{T})$ are linkable the algorithm performs the following steps:*

1. *Run the algorithm to merge $\sigma_\mathcal{S}$ and $\sigma_\mathcal{T}$;*
2. *If merging is possible return `true`, otherwise `false`.*

The construction 10 is elegantly simple, yet fulfils all security goals (all but unforgeability even perfectly), and is therefore useable in practice.

$merge\mathcal{RSS}$ is mergeable and thus linkable. The Link algorithm of $merge\mathcal{RSS}$ was black-box constructed based on a correct Merge following the idea for the property of linkability described in Sec. 14.4.

Note, using the Link algorithm one could think that it allows to offer a naïve instantiation of Proof and corresponding Judge: A Signer could retain all message-signature pairs and release a linkable message-signature pair as a proof for which the judge based on the linkability and the fact that the message in the proof contains more original blocks will appoint the Sanitizer as accountable. However, the fact that that proof must always contain more information than what what the verifier knows this would break privacy and transparency.

Note, in $merge\mathcal{RSS}$ it is made explicitly clear that the scheme will not detect the transitive closure of the updates as forgeries. If the Signer wants to disallow such a 'transitive update merging', the construction has to deploy accumulators which also update the witnesses, e.g., [104]. However, this requires a new security model, which renders existing constructions insecure. It was of utmost desire for the thesis to devise all proposed properties, especially those around the merge functionality, to avoid such a huge impact on existing works. As a consequence, the merge functionality must allow 'updates' just like the RSS does.

14.12.3. $merge\mathcal{RSS}$ runtime and complexity

The $merge\mathcal{RSS}$ construction was instantiated on a smart card. Please see Sec. 14.15.6 on page 430 for details on the runtime on a given smart card.

14.12.4. $merge\mathcal{RSS}$ security proof

Note, all reductions are tight, i.e., there is no reduction loss.

Theorem 38 : *merge$\mathcal{RSS}$* Construction 10 is Unforgeable.

Construction 10 is unforgeable, if the underlying accumulator is strongly collision-resistant.

Proof 38

Note, tag collisions are not considered as they only appear with negligible probability. $S^* \subseteq S_\tau$ for some signed τ is not a forgery, but a redaction. Let $\mathcal{A}$ denote the adversary winning the unforgeability game. The winning forgery must fall into exactly one of the following categories:

Case 1: $\mathcal{S}^* \nsubseteq \mathcal{S}_{\tau^*}$, and τ^* was used as a tag by Sign
Case 2: $\mathcal{S}^*$ verifies, and τ^* was never used as a tag by Sign

Each case leads to a contradiction about the security of underlying primitive: The accumulator is assumed to be strongly collision-resistant. This is shown by using the adversary $\mathcal{A}$ to build an adversary $\mathcal{B}$ that breaks the property of strong collision-resistance.

Case 1. *In this case, an element v^* is not been returned by the* Proof*-oracle for the accumulator a, but is contained in $\mathcal{S}^*$. One breaks the strong collision-resistance of the underlying accumulator by letting $\mathcal{B}$ use $\mathcal{A}$ as a black-box:*

1. *$\mathcal{B}$ receives $\mathsf{pk}_{\mathsf{ACC}}$ from the challenger*
2. *$\mathcal{B}$ requests an accumulator a for $\emptyset$*
3. *$\mathcal{B}$ receives a from its own challenger*
4. *$\mathcal{B}$ forwards $\mathsf{pk} = (\mathsf{pk}_{\mathsf{ACC}}, a)$ to $\mathcal{A}$*
5. *For each query to the signing oracle, $\mathcal{B}$ answers it honestly: it draws τ honestly and uses the* Proof*-oracle provided to get a witness for each $v_j \in \mathcal{S}_i$ queried, with τ concatenated as the label. Also, $\mathcal{B}$ gets a proof for τ*
6. *For each call to the* Update*-oracle, $\mathcal{B}$ uses its* Proof*-oracle provided to get a witness for each $v_j \in \mathcal{S}_i$ queried, with τ concatenated as the label*
7. *Eventually, $\mathcal{A}$ outputs a pair (S^*, σ^*)*

8. *$\mathcal{B}$ looks for (v^*, p^*), v^* not queried to* Proof*, in (S^*, σ^*) and returns them*

In other words, there exists an element $v^* \in \mathcal{S}^*$ with a corresponding witness p^*. If v^* has not been asked to the Proof*-oracle, $\mathcal{B}$ breaks the collision-resistance of the underlying accumulator by outputting (v^*, p^*). This happens with the same probability as $\mathcal{A}$ breaks unforgeability in case 1. Hence, the reduction is tight.*

Case 2. *In case 2, the tag τ^* has not been accumulated. One breaks the strong collision-resistance of the underlying accumulator by letting $\mathcal{B}$ use $\mathcal{A}$:*

1. *$\mathcal{B}$ receives $\mathsf{pk}_{\mathsf{ACC}}$ from the challenger*
2. *$\mathcal{B}$ requests an accumulator a for $\emptyset$*
3. *$\mathcal{B}$ forwards $\mathsf{pk} = (\mathsf{pk}_{\mathsf{ACC}}, a)$ to $\mathcal{A}$*
4. *For each query to the signing oracle, $\mathcal{B}$ answers it honestly: it draws τ honestly and uses the* Proof*-oracle provided to get a witness for each $v_j \in \mathcal{S}_i$ queried, with τ concatenated as the label. Also, $\mathcal{B}$ gets a proof for τ*
5. *For calls to the* Update*-oracle, $\mathcal{B}$ uses its* Proof*-oracle provided to get a witness for each $v_j \in \mathcal{S}_i$ queried, with τ concatenated as the label*
6. *Eventually, $\mathcal{A}$ outputs a pair (S^*, σ^*, τ^*)*
7. *$\mathcal{B}$ returns (p^*_τ, τ^*). Both is contained in σ^**

In other words, there exists an element $\tau^* \in \sigma^*$ with a corresponding witness p^*_τ, as otherwise σ^* would not verify. One knows that τ^* was not queried to Proof*, because otherwise it would be case 1. This happens with the same probability as $\mathcal{A}$ breaks the unforgeability in case 2. Note, one can ignore additional elements here. Again, the simulation is perfect. Perfect means that the property holds in an information-theoretical sense.* □

Theorem 39 : *merge$\mathcal{RSS}$* from Construction 10 is Merge Private and Transparent.

Construction 10 is merge private and merge transparent.

Proof 39

The distributions of merged and freshly signed signatures are equal. In other words, the distributions are the same. This implies that Construction 10 is perfectly merge private and perfectly merge transparent. Perfect here means the property holds in an information-theoretical sense. □

Theorem 40 : *merge$\mathcal{RSS}$* from Construction 10 is Transparent and Private.

Construction 10 is transparent and private.

Proof 40

As the number of proofs only depends on n, which are also deterministically generated, without taking existing proofs into account, an adversary has zero advantage on deciding how many additional proofs have been generated. Moreover, redacting only removes elements and proofs from the signatures. Hence, fresh and redacted signatures are distributed identically. Perfect transparency, and therefore also perfect privacy, is implied. Perfect here means the property holds in an information-theoretical sense. □

Theorem 41 : *merge$\mathcal{RSS}$* from Construction 10 is Update Private and Transparent.

Construction 10 is update private and update transparent.

Proof 41

The distributions of updated and freshly signed signatures are equal. In other words, the distributions are the same. This implies that Construction 10 is perfectly update private and perfectly update transparent. Perfect here means the property holds in an information-theoretical sense. □

14.12.5. *merge*$\mathcal{RSS}$ integrity offer and visualisation

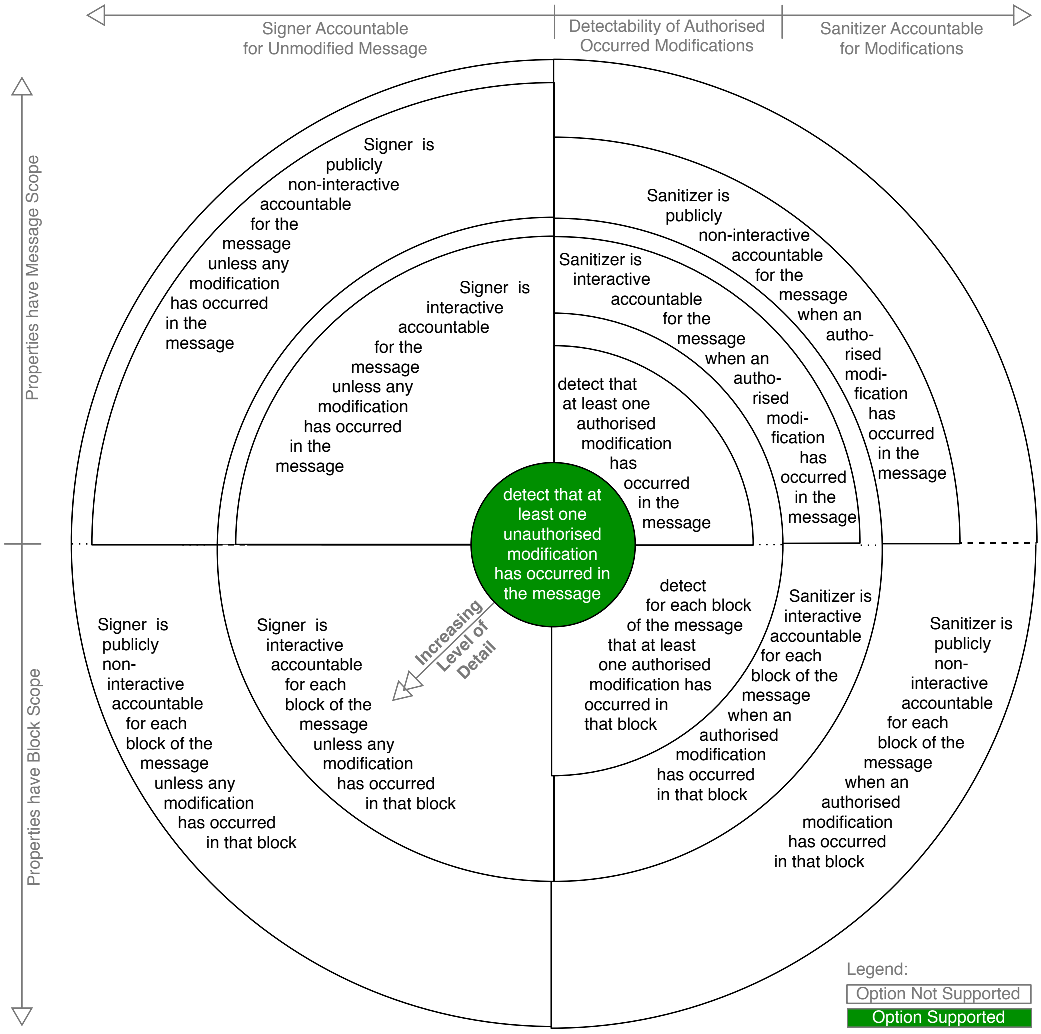

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an authorised modification has occurred in the message
Sanitizer is interactive accountable for the message when an authorised modification has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Increasing Level of Detail
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
□ Option Not Supported
🟩 Option Supported

Figure 117. *merge*$\mathcal{RSS}$ offers ACA – UCD – INT integrity, because Verifiers can detect always only unauthorized subsequent modifications as there is no differentiation between an original and a message modified with authorization inside the transparent *merge*$\mathcal{RSS}$ that would allow to hold a Sanitizer accountable

merge$\mathcal{RSS}$ is a redactable signature scheme for an ordered list. It offers transparency (and therefore privacy). It is an RSS that allows for public redactions, public merge and Signer updates. Due to transparency and the public redaction and merge operations this yields no detection capability for a Verifier for an occurred authorized subsequent modification. However, being mergeable *merge*$\mathcal{RSS}$ at the same time becomes linkable. This, however, does not yield an option for the Signer to generate an interactive proof that a message was redacted as this would break privacy, as discussed briefly at the end of Sec. 14.12.2. Thus *merge*$\mathcal{RSS}$ classifies as giving ACA – UCD – INT integrity and the Signer stays accountable for all messages that contain only authorized modifications. Fig. 117 visualises this.

14.13. *pubacc$\mathcal{RSS}$* Concept: Make the redact operation key dependent to gain accountability

The *pubacc$\mathcal{RSS}$*[855] scheme gains accountability by implementing the algorithms and provably achieving the security properties given for an ARSS in Sec. 14.1. The basic idea behind the ARSS framework is to inherit accountability from an SSS to make any RSS accountable. This is achieved by signing the original message m and the corresponding redactable signature σ_{RSS} of that m using the SSS. While m and σ_{RSS} can be changed publicly by redaction using the RSS, the outer SSS signature can only be updated by dedicated and accountable Sanitizers using the SSS. The construction of *pubacc$\mathcal{RSS}$* is elegantly simple.

Depending on the properties of the underlying building blocks, the resulting ARSS achieves public accountability or transparency: *pubacc$\mathcal{RSS}$* as presented next is a non-interactive publicly accountable RSS ($\mathrm{ACA} - \geq 1\mathrm{CD} - \mathrm{PUB}$). This is because this thesis, following the legally imposed requirements, strives to achieve non-interactive public accountability (PUB). But if the underlying building block is chosen without offering PUB-level accountability, the same can be used to construct a transparent ARSS ($\mathrm{ACA} - \geq 1\mathrm{CD} - \mathrm{INT}$). The former allows to reproducibly derive the accountable party without knowing any secrets and without any additional protocol interaction.

14.13.1. *pubacc$\mathcal{RSS}$* extended security properties

An accountable RSS (ARSS) is secure, if it is correct, unlinkable, unforgeable, Sanitizer-unforgeable, private, Sanitizer-accountable and Signer-accountable. The redactable signature *pubacc$\mathcal{RSS}$* for lists achieves the following security properties (**bold** indicates a new property introduced in this thesis):

- Unforgeability according to Definition 172, and
- Standard Privacy following *Brzuska et al.* according to Definition 174, and additionally
- **Non-Interactive Public Accountability** according to Definition 204 (achieved by *pubacc$\mathcal{RSS}$*),
  - alternatively, Transparency (with an alternate instantiation).

As commonly stated in this thesis, a scheme is only able to offer either transparency or public accountability as they are mutually exclusive. The choice depends on the use case. In the remainder, the focus is on the public form of accountability due to the legal benefits (see for example the brief discussion in Sec. 6.4.5).

14.13.2. *pubacc$\mathcal{RSS}$* algorithm

The *pubacc$\mathcal{RSS}$* algorithm implements all the algorithms of the ARSS framework presented previously in Sec. 14.1.

14.13.3. *pubacc$\mathcal{RSS}$* correctness properties

The usual correctness properties for RSS from Sec. 11.11 must hold.

14.14. *pubacc$\mathcal{RSS}$* cryptographic instantiation

This result has been published, as part of the joint work with *K. Samelin*, *J. Posegga* and *H. de Meer* [388], at ARES 2015 (see Appendix A publication n° 21).

Within the formal framework for an accountable redactable signature ARSS (see Sec. 14.1) *pubacc$\mathcal{RSS}$* is constructed. As in the construction ARSS.Judge is directly built upon SSS.Judge the *pubacc$\mathcal{RSS}$* construction inherits from the properties of the SSS used as an underlying building block. Thus, with the right SSS underneath, the *pubacc$\mathcal{RSS}$* construction achieves non-interactive public (PUB) accountability. An alternate SSS could be used to construct an ARSS that allows to offer transparency

[855] The construction and security proofs for *pubacc$\mathcal{RSS}$* are partly published in the joint work with *K. Samelin* [388] published at the ARES conference in 2015 (see Appendix A publication n° 21).

(INT). For the ARSS to exhibit transparency both building blocks, RSS and SSS, are required to be transparent. The construction exclusively covers messages represented as ordered lists of blocks and protects the order as structural integrity. However, as the proofs do not depend on this structure, this thesis assumed that these ideas can be extended easily to more complex data-structures.

14.14.1. $pubacc\mathcal{RSS}$ construction

Construction 11 : $pubacc\mathcal{RSS}$

Let $\mathsf{RSS} = (\mathsf{RSS.KGen}_{sig}, \mathsf{RSS.Sign}, \mathsf{RSS.Redact}, \mathsf{RSS.Verify})$ be a secure redactable signature scheme. Let $\mathsf{SSS} = (\mathsf{SSS.KGen}_{sig}, \mathsf{SSS.KGen}_{san}, \mathsf{SSS.Sign}, \mathsf{SSS.Sanit}, \mathsf{SSS.Verify}, \mathsf{SSS.Proof}, \mathsf{SSS.Judge})$ be a secure sanitizable signature scheme. For brevity of the representation below, the security parameter 1^λ is just given directly to key generation algorithms and it is assumed that all other algorithm needing it can always derive it from the keys they are provided. Also the modification policy ADM is assumed to be always reconstructable from any valid signature-message pair and thus is only explicitly listed as input for Sign.

Then an accountable redactable signature $pubacc\mathcal{RSS} = (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Redact}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$ is as follows:

Key Generation: *KGen_{sig} generates on input of the security parameter λ a key pair. It consists of a key pair for an RSS: $(\mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{sk}^{\mathsf{RSS}}_{sig}) \leftarrow \mathsf{RSS.KGen}_{sig}(1^\lambda)$ and for an SSS: $(\mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{sk}^{\mathsf{SSS}}_{sig}) \leftarrow \mathsf{SSS.KGen}_{sig}(1^\lambda)$. It returns $((\mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{RSS}}_{sig}), (\mathsf{sk}^{\mathsf{SSS}}_{sig}, \mathsf{sk}^{\mathsf{RSS}}_{sig}))$. Analogously, KGen_{san} returns a pair $(\mathsf{pk}^{\mathsf{SSS}}_{san}, \mathsf{sk}^{\mathsf{SSS}}_{san}) \leftarrow \mathsf{SSS.KGen}_{san}(1^\lambda)$.*

Signing: *Sign, on input of m, sk_{sig}, and pk_{san} computes $\sigma_{\mathsf{RSS}} \leftarrow \mathsf{RSS.Sign}(m, \mathsf{sk}^{\mathsf{RSS}}_{sig})$. It lets $\sigma_{\mathsf{SSS}} \leftarrow \mathsf{SSS.Sign}((m, \sigma_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}), \mathsf{sk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}, \mathsf{ADM})$, where $\mathsf{ADM} = (\{1,2\}, 5)$. It returns $\sigma = (\sigma_{\mathsf{RSS}}, \sigma_{\mathsf{SSS}})$.*

Redacting: *Redact, on input of message m, modification instructions MOD, a signature $\sigma = (\sigma_{\mathsf{RSS}}, \sigma_{\mathsf{SSS}})$, checks if σ is valid using Verify. If not, it returns $\bot$. If all checks pass, it generates $(m', \sigma'_{\mathsf{RSS}}) \leftarrow \mathsf{RSS.Redact}(m, \mathsf{MOD}, \sigma_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig})$ and $(m'', \sigma'_{\mathsf{SSS}}) \leftarrow \mathsf{SSS.Sanit}((m, \sigma_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}), \{(1, m'), (2, \sigma'_{\mathsf{RSS}})\}, \sigma_{\mathsf{SSS}}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{sk}^{\mathsf{SSS}}_{san})$. It finally returns $(m', (\sigma'_{\mathsf{RSS}}, \sigma'_{\mathsf{SSS}}))$.*

Verification: *Algorithm Verify on input of a message m a signature $\sigma = (\sigma_{\mathsf{RSS}}, \sigma_{\mathsf{SSS}})$ and public keys pk_{sig} and pk_{san}, first checks if σ_{RSS} is a valid signature using $\mathsf{RSS.Verify}(m, \sigma_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig})$. It then checks whether σ_{SSS} is valid running $\mathsf{SSS.Verify}((m, \sigma_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}), \sigma_{\mathsf{SSS}}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san})$. If all checks are valid, return `true`. Otherwise it returns `false`.*

Proof: *Proof on input of $\sigma = (\sigma_{\mathsf{RSS}}, \sigma_{\mathsf{SSS}})$, m, pk_{san}, sk_{sig}, and $\{m_i, (\sigma_{\mathsf{RSS},i}, \sigma_{\mathsf{SSS},i}) \mid i \in \mathbb{N}\}$, it first checks whether σ is valid. If not, it returns $\bot$. Else, it returns $\pi \leftarrow \mathsf{SSS.Proof}(\mathsf{sk}^{\mathsf{SSS}}_{sig}, (m, \sigma_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}), \sigma_{\mathsf{SSS}}, \{((m_i, \sigma_{\mathsf{RSS},i}, \mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}), \sigma_{\mathsf{SSS},i}) \mid i \in \mathbb{N}\}, \mathsf{pk}^{\mathsf{SSS}}_{san})$.*

Judge: *Judge on input of $m, \sigma = (\sigma_{\mathsf{RSS}}, \sigma_{\mathsf{SSS}}), \mathsf{pk}_{sig}, \mathsf{pk}_{san}$ and π first checks whether σ is valid. If not, it outputs $\bot$. Else, it outputs $\mathsf{SSS.Judge}((m, \sigma_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}), \sigma_{\mathsf{SSS}}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{san}, \pi)$.*

14.14.2. $pubacc\mathcal{RSS}$ runtime and complexity

The runtime and complexity follows as the combined runtimes of the underlying SSS and RSS.

14.14.3. $pubacc\mathcal{RSS}$ security proof

In the following the proof of security and accountability for the construction of $pubacc\mathcal{RSS}$ is presented.

Theorem 42 : $pubacc\mathcal{RSS}$ from Construction 11 is Secure and Accountable.

> *If* SSS *is a secure sanitizable signature scheme, while* RSS *is a secure redactable signature scheme, the resulting construction $pubacc\mathcal{RSS}$ is a secure accountable redactable signature scheme (*ARSS*).*

Note again: This thesis does neither treat transparency nor public accountability as an essential security requirement for RSS or SSS in general. The schemes shall be flexible enough to trade probative value against an increased privacy, as it highly depends on the use-case which security property is desired.

Suitable instantiations for the non-interactive public accountable SSS include, e.g., $pubacc\mathcal{SSS}$ (see Sec. 13.4 and Sec. 13.5 for design and proofs) or the scheme by *Fleischhacker, Krupp, Malavolta, Schneider, Schröder, and Simkin* [196].

To prove Theorem 42 it is needed to show that construction Sec. 14.14.1 is unforgeable, Signer-accountable, Sanitizer-accountable, unlinkable, Sanitizer-unforgeable, and private. The proof for $pubacc\mathcal{RSS}$ being publicly accountable is given last in Theorem 43. Note, transparency and public accountability are mutually exclusive; for transparency it is required that the SSS and the RSS are transparent, while for public accountability, only the SSS needs to be publicly accountable.

Proof 42

Unforgeability. *According to the definition of unforgeability for* RSS*, the only cases where the adversary $\mathcal{A}$ wins the game is when it comes up with a valid signature σ^* for a new message m^* which has not been signed under the given public keys. In the following this case is reduced to the unforgeability of the underlying* SSS*. Let $\mathcal{A}$ be an algorithm which breaks the unforgeability of* ARSS*. One can then construct an algorithm $\mathcal{B}$ which uses $\mathcal{A}$ to break the unforgeability of the* SSS*:*

1. *$\mathcal{B}$ creates a key pair of an* RSS*, i.e.,* $(pk_{sig}^{RSS}, sk_{sig}^{RSS}) \leftarrow RSS.KGen_{sig}(1^\lambda)$
2. *$\mathcal{B}$ receives both public keys, i.e.,* pk_{sig}^{SSS} *and* pk_{san}^{SSS} *of the* SSS *to forge.*
3. *$\mathcal{B}$ passes* pk_{sig}^{SSS}, pk_{san}^{SSS}, *and* pk_{sig}^{RSS} *to $\mathcal{A}$ as the* ARSS*'s public keys.*
4. *For every query made, $\mathcal{B}$ uses its own oracles and keys to simulate $\mathcal{A}$'s environment.*
5. *Eventually, $\mathcal{A}$ returns a message/signature pair (m^*, σ^*). If this tuple verifies under the given public keys and has not been queried to the signing or redacting oracles, m^* is also a fresh message for the underlying* SSS*. $\mathcal{B}$ can therefore return $((m^*, \sigma^*_{RSS}, pk_{sig}^{RSS}, pk_{sig}^{SSS}, pk_{san}^{SSS}), \sigma^*_{SSS})$, as its own forgery attempt. The probability that $\mathcal{B}$ wins the* SSS *unforgeability game is the same as $\mathcal{A}$'s, as the simulation is perfect and the message m^* is fresh.*

Sanitizer-Unforgeability. *Both cases where the adversary $\mathcal{A}$ wins the game require it to present a valid signature σ^* for a message m^* which has never been endorsed by the* Signer *under the given* pk^**. In the following a reduction for each case is provided.*

Case 1: If pk^* *was never queried, then one derives an algorithm $\mathcal{B}$ which uses $\mathcal{A}$ internally to break the immutability requirement of the underlying* SSS*. $\mathcal{B}$ proceeds as follows:*

1. *$\mathcal{B}$ creates a key pair of an* RSS*, i.e.,* $(pk_{sig}^{RSS}, sk_{sig}^{RSS}) \leftarrow RSS.KGen_{sig}(1^\lambda)$
2. *$\mathcal{B}$ receives the* Signer*'s public keys, i.e.,* pk_{sig}^{SSS} *of the* SSS *to forge.*
3. *$\mathcal{B}$ passes* pk_{sig}^{SSS} *and* pk_{sig}^{RSS} *to $\mathcal{A}$ as the* ARSS*'s public keys.*
4. *For every query made, $\mathcal{B}$ uses its own oracles and keys to simulate $\mathcal{A}$'s environment.*

5. *Eventually, $\mathcal{A}$ returns a message/signature pair $(m^*, \sigma^*, \mathsf{pk}^*)$, where $\sigma^* = (\sigma^*_{\mathsf{RSS}}, \sigma^*_{\mathsf{SSS}})$. By assumption, pk^* is also a fresh public key of the* Sanitizer *of the underlying* SSS*. If this tuple verifies, $\mathcal{B}$ can therefore return $(m', \sigma^*_{\mathsf{SSS}}, \mathsf{pk}^*)$, as $\sigma^* = (\sigma^*_{\mathsf{RSS}}, \sigma^*_{\mathsf{SSS}})$ where $m' = (m^*, \sigma^*_{\mathsf{RSS}}, \mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{pk}^*)$ as its own forgery attempt. The probability that $\mathcal{B}$ wins the* SSS *immutability game is therefore the same as $\mathcal{A}$'s advantage of winning the* Sanitizer*-unforgeability game of the* ARSS*, as the simulation is perfect.*

Case 2: When a signed m^* should not have been derivable by redaction, i.e., $m^* \notin \bigcup_{i=1}^{q} span_{\vdash}(m_i)$ for every query to the signing oracle with pk^*, then one derives an algorithm $\mathcal{B}$ which uses $\mathcal{A}$ internally to break the unforgeability requirement of the underlying RSS*. $\mathcal{B}$ proceeds as follows.*

1. *$\mathcal{B}$ creates a key pair of an* SSS*, i.e., $(\mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{sk}^{\mathsf{SSS}}_{sig}) \leftarrow \mathsf{SSS.KGen}_{sig}(1^\lambda)$*

2. *$\mathcal{B}$ receives the public keys, i.e., $\mathsf{pk}^{\mathsf{RSS}}_{sig}$ of the* RSS *to forge.*

3. *$\mathcal{B}$ passes $\mathsf{pk}^{\mathsf{SSS}}_{sig}$ and $\mathsf{pk}^{\mathsf{RSS}}_{sig}$ to $\mathcal{A}$ as* ARSS*'s public keys.*

4. *For every query made, $\mathcal{B}$ uses its own oracles and keys to simulate $\mathcal{A}$'s environment.*

5. *Eventually, $\mathcal{A}$ returns $(m^*, \sigma^*, \mathsf{pk}^*)$, where $\sigma^* = (\sigma^*_{\mathsf{RSS}}, \sigma^*_{\mathsf{SSS}})$. This tuple verifies, and, by assumption, m^* is not derivable by any query made. Hence, m^* is also a forgery of the* RSS*. $\mathcal{B}$ can therefore return $(m^*, \sigma^*_{\mathsf{RSS}})$ as its own forgery attempt. The probability that $\mathcal{B}$ wins the* RSS *unforgeability game is therefore the same as $\mathcal{A}$'s advantage of winning the* Sanitizer*-unforgeability game of the* ARSS*.*

Privacy. *According to the definition of privacy, the adversary $\mathcal{A}$ has to guess a bit. One can use $\mathcal{A}$ to either break privacy of the* SSS *or the* RSS*. $\mathcal{B}$ uses $\mathcal{A}$ as a block-box:*

1. *$\mathcal{B}$ tosses a coin $b \leftarrow \{0, 1\}$.*

2. *If $b = 0$, $\mathcal{B}$ creates both key pairs of an* SSS*, i.e., $(\mathsf{pk}^{\mathsf{SSS}}_{sig}, \mathsf{sk}^{\mathsf{SSS}}_{sig}) \leftarrow \mathsf{SSS.KGen}_{sig}(1^\lambda)$ and $(\mathsf{pk}^{\mathsf{SSS}}_{san}, \mathsf{sk}^{\mathsf{SSS}}_{san}) \leftarrow \mathsf{SSS.KGen}_{san}(1^\lambda)$ and receives* RSS*'s public key, $\mathsf{pk}^{\mathsf{RSS}}_{sig}$.*

3. *If $b = 1$, $\mathcal{B}$ creates a key pair of an* RSS*, i.e., $(\mathsf{pk}^{\mathsf{RSS}}_{sig}, \mathsf{sk}^{\mathsf{RSS}}_{sig}) \leftarrow \mathsf{RSS.KGen}_{sig}(1^\lambda)$ and receives* SSS*'s public keys, i.e., $\mathsf{pk}^{\mathsf{SSS}}_{san}$ and $\mathsf{pk}^{\mathsf{SSS}}_{sig}$.*

4. *$\mathcal{B}$ passes $\mathsf{pk}^{\mathsf{SSS}}_{sig}$, $\mathsf{pk}^{\mathsf{SSS}}_{san}$ and $\mathsf{pk}^{\mathsf{RSS}}_{sig}$ to $\mathcal{A}$ as the* ARSS*'s public keys.*

5. *For every query made, $\mathcal{B}$ uses its own oracles/keys to simulate $\mathcal{A}$'s environment.*

6. *Eventually, $\mathcal{A}$ returns its own guess b^*. $\mathcal{B}$ returns b^* as its own guess. What is the probability that $\mathcal{B}$'s guess is correct? Assuming that $\mathcal{A}$'s advantage against privacy is ϵ, while at least one of the underlying building blocks must be non-private (this does not introduce additional information), $\mathcal{B}$'s advantage against privacy of either the* RSS *or the* SSS *is $\geq \frac{\epsilon}{2}$: the simulation for the correct case was done with probability of $\frac{1}{2}$.*

Unlinkability. *In the definition of unlinkability, the adversary $\mathcal{A}$ has to guess a bit. Again, one can use $\mathcal{A}$ to either break unlinkability of the* SSS *or the* RSS*. As usual, one uses $\mathcal{A}$ as a black-box in an algorithm $\mathcal{B}$:*

1. *$\mathcal{B}$ tosses a coin $b \leftarrow \{0, 1\}$.*

2. *If $b = 0$, $\mathcal{B}$ creates a* Sanitizer *key pair of an* SSS*, i.e., $(\mathsf{pk}^{\mathsf{SSS}}_{san}, \mathsf{sk}^{\mathsf{SSS}}_{san}) \leftarrow \mathsf{SSS.KGen}_{san}(1^\lambda)$.*

3. *If $b = 1$, $\mathcal{B}$ receives $(\mathsf{pk}^{\mathsf{SSS}}_{san})$ from its own challenger.*

4. *$\mathcal{B}$ passes $\mathsf{pk}^{\mathsf{SSS}}_{san}$ to $\mathcal{A}$ as the* Sanitizer*'s public key.*

5. *For every query made, $\mathcal{B}$ uses its own oracles/keys to simulate $\mathcal{A}$'s environment accordingly.*

6. *Eventually, $\mathcal{A}$ returns its guess b^*. $\mathcal{B}$ returns b^* as its own guess. What is the probability that $\mathcal{B}$'s guess is correct? Assuming that $\mathcal{A}$'s advantage against unlinkability is ϵ and not negligible, while at least one of the underlying building blocks must be non-unlinkability (this does not introduce additional information), $\mathcal{B}$'s advantage against unlinkability of either the RSS or the SSS is $\geq \frac{\epsilon}{2}$, as the simulation for the correct case was done with probability of exactly $\frac{1}{2}$.*

Transparency. *For transparency, the adversary $\mathcal{A}$ has to also guess a bit. One can simply use $\mathcal{A}$ to either break transparency of the SSS or the RSS. Once more, one uses $\mathcal{A}$ as a black-box in an algorithm $\mathcal{B}$:*

1. *$\mathcal{B}$ tosses a coin $b \leftarrow \{0, 1\}$.*
2. *If $b = 0$, $\mathcal{B}$ creates both key pairs of an SSS, i.e., $(pk^{SSS}_{sig}, sk^{SSS}_{sig}) \leftarrow SSS.KGen_{sig}(1^\lambda)$ and $(pk^{SSS}_{san}, sk^{SSS}_{san}) \leftarrow SSS.KGen_{san}(1^\lambda)$ and receives RSS's public key, pk^{RSS}_{sig}.*
3. *If $b = 1$, $\mathcal{B}$ creates a key pair of an RSS, i.e., $(pk^{RSS}_{sig}, sk^{RSS}_{sig}) \leftarrow RSS.KGen_{sig}(1^\lambda)$ and receives the public keys of the SSS, i.e., pk^{SSS}_{san} and pk^{SSS}_{sig}.*
4. *$\mathcal{B}$ passes pk^{SSS}_{sig}, pk^{SSS}_{san} and pk^{RSS}_{sig} to $\mathcal{A}$ as the ARSS's public keys.*
5. *For every query made, $\mathcal{B}$ uses its own oracles/keys to simulate $\mathcal{A}$'s environment.*
6. *Eventually, $\mathcal{A}$ returns its own guess b^*. $\mathcal{B}$ returns b^* as its own guess. Assuming that $\mathcal{A}$'s advantage against transparency is ϵ, while at least one of the underlying building blocks must be non-transparent (once more: No additional information is introduced in this construction), $\mathcal{B}$'s advantage against transparency of either the RSS or the SSS is $\geq \frac{\epsilon}{2}$, as the simulation for the correct case was done with probability $\frac{1}{2}$.*

Signer-Accountability. *According to the definition of Signer-accountability, the adversary $\mathcal{A}$ has to generate $pk^*, \pi^*, m^*, \sigma^*$ which makes Judge decide wrongly, i.e., it outputs `San` while (m^*, pk^*) has never been queried to Redact. One uses $\mathcal{A}$ as a black-box in an algorithm $\mathcal{B}$ to break the Signer-accountability of the underlying SSS:*

1. *$\mathcal{B}$ receives pk^{SSS}_{san} from its own challenger.*
2. *$\mathcal{B}$ passes pk^{SSS}_{san} to $\mathcal{A}$ as the Sanitizer's public key.*
3. *For every query made, $\mathcal{B}$ uses its own Sanit-oracle to simulate $\mathcal{A}$'s environment.*
4. *Eventually, $\mathcal{A}$ returns $(pk^*, \pi^*, m^*, \sigma^*)$.*
   $\mathcal{B}$ returns $(pk^{SSS*}_{sig}, \pi^*, (m^*, \sigma^*_{RSS}, pk^{RSS*}_{sig}, pk^{SSS*}_{sig}, pk^{SSS}_{san}), \sigma^*_{SSS})$, where $\sigma^* = (\sigma^*_{RSS}, \sigma^*_{SSS})$ and $pk^* {=} (pk^{SSS*}_{sig}, pk^{RSS*}_{sig})$. As one only forwards queries, and either m^* or pk^* are "fresh", the probability that $\mathcal{B}$ wins the SSS Signer-accountability game is therefore the same as $\mathcal{A}$'s advantage of winning the Signer-unforgeability game of the ARSS.

Sanitizer-Accountability. *The adversary $\mathcal{A}$ has to generate pk^*, m^*, σ^* which makes Proof output a proof π which makes Judge decide wrongly, i.e., Judge outputs `Sig` while (m^*, pk^*) has never been queried to Sign. One uses $\mathcal{A}$ as a black-box in an algorithm $\mathcal{B}$ again to break the Sanitizer-accountability of the underlying SSS:*

1. *$\mathcal{B}$ creates a key pair of an RSS, i.e., $(pk^{RSS}_{sig}, sk^{RSS}_{sig}) \leftarrow RSS.KGen_{sig}(1^\lambda)$*
2. *$\mathcal{B}$ receives pk^{SSS}_{sig} from its own challenger.*
3. *$\mathcal{B}$ passes pk^{SSS}_{sig} and pk^{RSS}_{sig} to $\mathcal{A}$ as the Signer's public keys.*
4. *For every query made, $\mathcal{B}$ uses its own oracles and keys to simulate $\mathcal{A}$'s environment.*

5. *Eventually, $\mathcal{A}$ returns (pk^*, m^*, σ^*).*
   $\mathcal{B}$ returns $(pk^*_{san}, (m^*, \sigma^*_{RSS}, pk^{RSS}_{sig}, pk^{SSS}_{sig}, pk^*), \sigma^*_{SSS})$, where $\sigma^* = (\sigma^*_{RSS}, \sigma^*_{SSS})$. As one only forwards queries, and either m^* or pk^* are "fresh", the probability that $\mathcal{B}$ wins the SSS Sanitizer-accountability game is therefore the same as $\mathcal{A}$'s advantage of winning the Sanitizer-unforgeability game of the ARSS.

□

Theorem 43 : A secure message-level accountable ARSS can be non-interactive public accountable (PUB).

If SSS is a secure message-level non-interactive public accountable sanitizable signature scheme, while RSS is a secure redactable signature scheme, the resulting construction $pubacc\mathcal{RSS}$ is a secure message-level non-interactive public accountable (PUB) redactable signature scheme (ARSS).

Note, Theorem 43 speaks of message-level properties due to the ARSS framework (as described in Sec. 14.1.4), which offers only algorithms for message-level access to the judge. It is not a cryptographic limitation per se. It is out of the scope of this thesis to describe this for block-level as well.

Proof 43

According to the definition of public accountability, the adversary $\mathcal{A}$ has to generate pk^*, m^*, σ^* which makes Judge decide wrongly on an empty proof $\pi = \perp$, or the adversary came up with a new public key/message pair not queried. In both cases, one can reduce the security of ARSS to the security of the underlying SSS. In particular, $\mathcal{A}$ is used as a black-box in an algorithm $\mathcal{B}$ to break the public accountability of the underlying SSS:

1. *$\mathcal{B}$ creates RSS key pair $(pk^{RSS}_{sig}, sk^{RSS}_{sig}) \leftarrow RSS.KGen_{sig}(1^\lambda)$*
2. *$\mathcal{B}$ receives $(pk^{SSS}_{sig}, pk^{SSS}_{san})$ from its own challenger.*
3. *$\mathcal{B}$ forwards public keys to $\mathcal{A}$ as the ARSS's public keys.*
4. *For every query made, $\mathcal{B}$ uses its own oracles/keys to simulate $\mathcal{A}$'s environment.*
5. *Eventually, $\mathcal{A}$ returns (pk^*, m^*, σ^*). If pk^* only consists of one value, $\mathcal{B}$ returns $(pk^*, (m^*, \sigma^*_{RSS}, pk^{RSS}_{sig}, pk^{SSS}_{sig}, pk^*), \sigma^*_{SSS})$, where $\sigma^* = (\sigma^*_{RSS}, \sigma^*_{SSS})$ and $pk^* = pk^{SSS*}_{san}$. If pk^* consists of two public keys, $\mathcal{B}$ returns $(pk^{SSS*}_{sig}, (m^*, \sigma^*_{RSS}, pk^{RSS*}_{sig}, pk^{SSS*}_{sig}, pk^{SSS}_{san}), \sigma^*_{SSS})$, where $\sigma^* = (\sigma^*_{RSS}, \sigma^*_{SSS})$ and $pk^* = (pk^{RSS*}_{sig}, pk^{SSS*}_{sig})$. What is the probability that $\mathcal{B}$ wins its own game? If pk^* was never queried, but all public keys are signed by the SSS, the public accountability of the underlying SSS is obviously broken. In case Judge decides wrongly, then m^* was never queried to the corresponding oracle, also winning the public accountability of the underlying SSS. Hence, $\mathcal{B}$'s advantage equals the one of $\mathcal{A}$.*

Finally, from the proofs of Sanitizer-accountability and non-interactive public accountability one can follow that also the SSS properties towards detectability overtop a transparent RSS, e.g. if the RSS is transparent an non-interactively publicly accountable SSS will allow the combined *$pubacc\mathcal{RSS}$* to offer ACA – 1CD – PUB.

Theorem 44 : *$pubacc\mathcal{RSS}$* Construction 11 offer ACA – ≥1CD – PUB even if the underlying RSS does not

The ARSS from Construction 11 ($pubacc\mathcal{RSS}$) offers ACA – ≥1CD – PUB if the underlying SSS is secure and offers ACA – ≥1CD PUB (unforgeable, immutable, standard private and non-interactive public accountable) and if the underlying RSS is secure (unforgeable and standard private). In particular it is not required for the underlying RSS to offer ACA – ≥1CD – PUB, and any secure RSS achieves at least ACA – UCD integrity protection.

Proof 44 : Sketch

In the Construction Construction 11 the signature σ_{SSS} covers the full length of the message m after signature generation, as it is put into block number one, i.e. the message m_{SSS} given to SSS.Sign has five blocks: $m_{SSS} = m||\sigma_{RSS}||pk^{RSS}_{sig}||pk^{SSS}_{sig}||pk^{SSS}_{san}$. Any subsequent modification due to a redaction of a block inside m will need to adjust it to

m', it might not need to adjust σ_{RSS} depending on the RSS*. Obviously, even a transparent* RSS *will require to remove the redacted content from the message. Consequently, m' is provided as input to the algorithm* SSS.Sanit *during pubacc$\mathcal{RSS}$'s redaction algorithm. Thus, the subsequent modification is recorded by the* SSS *and is reflected in σ_{SSS}. Due to this, the detection of any subsequent modification can rest alone on the secure* SSS*. If the* SSS *achieves public non-interactive accountability the* Verifier *can corroborate the absence of unauthorized modifications when obtaining a valid verification outcome. Further, the* Verifier *can identify the absence of any authorized subsequent modification when the* Judge *on an empty proof identifies the* Signer *as accountable. Thus the scheme achieves ACA – ≥1CD – PUB.* □

14.14.4. *pubacc$\mathcal{RSS}$* integrity offer and visualisation

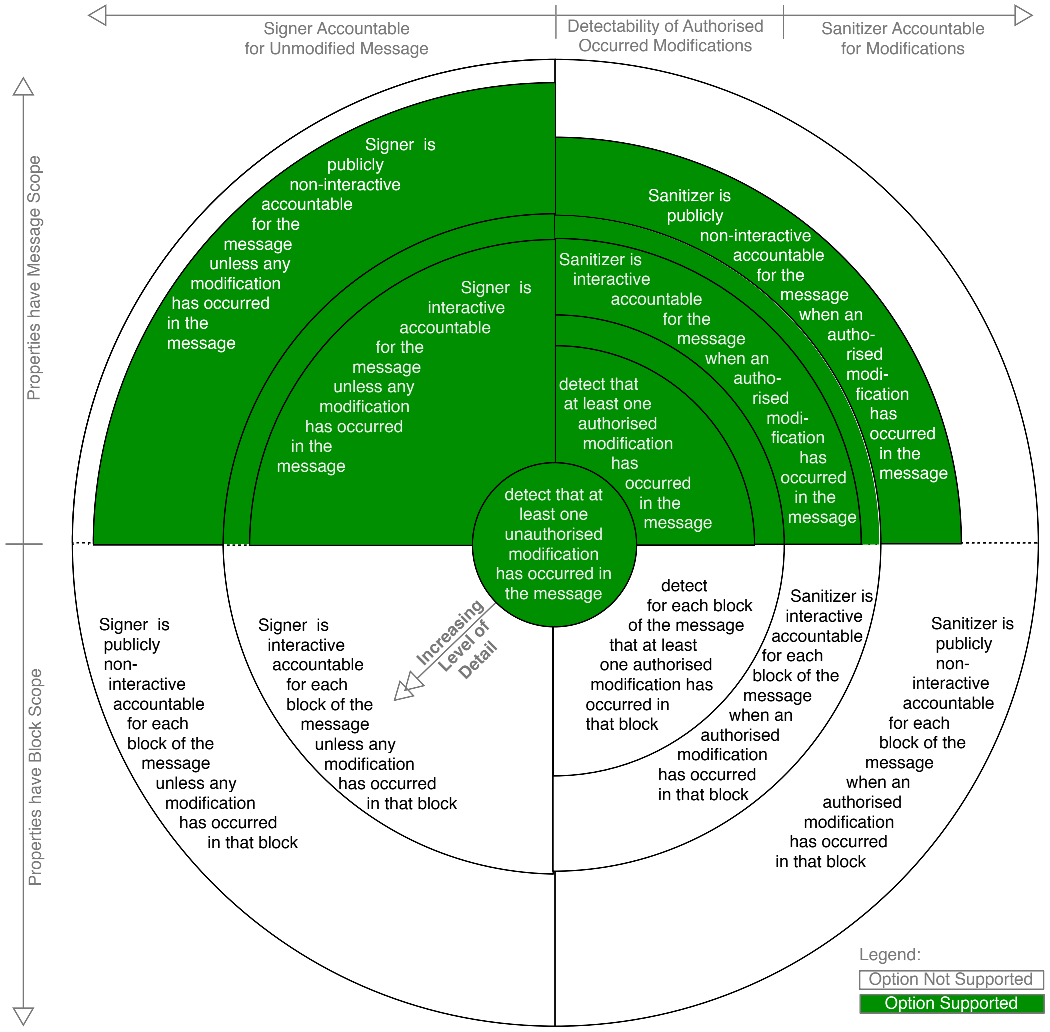

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Increasing Level of Detail
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Supported
Option Supported

Figure 118. *pubacc$\mathcal{RSS}$* offers ACA – 1CD – PUB integrity if based on an SSS that offers ACA – 1CD – PUB, because Verifiers are able to detect authorized subsequent redactions for which the redactor is accountable through its dedicated key pair

$pubacc\mathcal{RSS}$ is a message-level non-interactive public (PUB) accountable redactable signature scheme for ordered messages. It is an ARSS thus it allows no longer for public redactions, i.e. redaction is no longer an 'unkeyed' operation. To gain the accountability, the redacting entity needs to be endorsed by the Signer and is identified, like a Sanitizer in SSS, through its own key pair. $pubacc\mathcal{RSS}$'s new detection capability gained from non-interactive public accountability of the interwoven SSS results in the following: If $pubacc\mathcal{RSS}$ is based on an SSS that offers $\mathrm{ACA}-1\mathrm{CD}-\mathrm{PUB}$ $pubacc\mathcal{RSS}$ offers $\mathrm{ACA}-1\mathrm{CD}-\mathrm{PUB}$ integrity. If the RSS and the SSS is private it offers privacy. Fig. 118 visualises this.

14.15. Implementation of an RSS on a secure signature creation device (smart card)

A proof-of-concept implementations of the $merge\mathcal{RSS}$ RSS scheme on a smart card was made to identify if an RSS can meet Requirement 6[856]. The following section's results regarding the implementation got published as joint work with *S. Peters*, *K. Samelin*, *J. Posegga*, and *H. de Meer* at WISTP 2013 [395] (see Appendix A publication n$^{\mathrm{o}}$ 15). The $merge\mathcal{RSS}$ scheme's concept was described in detail in Sec. 14.11 and the cryptographic instantiation was presented in detail in Sec. 14.12. The scheme itself was published as joint work at ACNS 2014 [387] (see Appendix A publication n$^{\mathrm{o}}$ 20). The implementation presented next is based on the version of $merge\mathcal{RSS}$ jointly published as a technical report [393] (see Appendix A publication n$^{\mathrm{o}}$ 25) in 2012.

14.15.1. Smart card test environment

The test environment and test parameters and assumptions are the same as for SSS given in Sec. 13.10.1.

14.15.2. $merge\mathcal{RSS}$ on smart card

To recall: The core idea of $merge\mathcal{RSS}$ is to accumulate each block's hash into a single accumulated value, using a cryptographic accumulator (see Sec. 14.12.2). This accumulator value is signed with a standard signature scheme. Each time a block is accumulated, a witness that corresponds to the block is generated and an updated accumulator value is derived. Hence, the signed accumulator value is used to provide assurance that a block was signed given the Verifier knows the block and the corresponding witness. A redaction removes the block and its witness. Refer to Sec. 14.2.4 for details on the security model.

$merge\mathcal{RSS}$ requires the cryptographic accumulator ACC to be collision-resistant without trusted setup. Namely, the property of being without trusted setup is needed for the commitment property. The environment of the smart card facilitated for this proof-of-concept implementation was a standard commercially available smart card, namely a "SmartC@fé® Expert 4.x" from *Giesecke and Devrient* [206]. Consequently, this meant that no access to the underlying cryptographic hardware acceleration was possible from the standard software interface available for normal developers.[857] Next, this section presents the new accumulator scheme devised for use in this particular smart card scenario.

14.15.2.1. New smart-card-friendly accumulator with semi-trusted setup

Foremost, $merge\mathcal{RSS}$ requires the ACC's setup to hide certain values used for the parameter generation from untrusted parties, as knowledge allows efficient computation of collisions and thus forgeries of signatures. All known collision-resistant accumulators based on number theoretic assumptions either require a trusted third party, named the accumulator manager [43, 312], or they are very inefficient. As also stated, the trusted third party used for setup of the ACC must be trusted not to generate collisions to forge signatures. However, existing schemes without trusted third party are not efficiently implementable, e.g., the scheme without trusted setup introduced by *Sander* requires a modulus size of $\gg 40,000\,\mathrm{Bit}$ [429].

[856] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

[857] See also *Tews and Jacobs* [465] for a discussion.

Contrary, a smart card implementation for the digital signature scheme would usually already rely on an existing PKI infrastructure for the keys. In other words, for the Signer's public key certificate there would already be a trusted third party, denoted TTP, that signs the Signer's public key. The following construction thus assumes that TTP which signs a participant's public key uses an RSA-based scheme. It further assumes that TTP also runs the ACC setup. Thereby, TTP already has a secret: the standard RSA modulus $n = pq$, $p, q \in \mathbb{P}$. A naive construction of an accumulator setup could re-use n as the RSA-accumulator's modulus [43]. However, TTP could add new elements without detection. Thus, it would not be a semi-trusted, but a fully trusted setup. To achieve semi-trusted setup the proposed construction multiplies the modulus with further primes, called 'blinding primes'.

Joint generation of the modulus ensures semi-trusted setup, as long as the TTP and the Signer do not collude. By further multiplying the original modulus with 'blinding primes' during signing the scheme prohibits both, the trusted third party (TTP) and Signer, from finding collisions — unless they collude. Hence, this setting got called semi-trusted setup. This trade-off still requires a trusted third party for the setup, but inhibits the TTP from forging signatures generated by a Signer.

Note, without the need to jointly compute a modulus of unknown factorisation there is also no need for any protocol runs. Thus, keys can be generated off-line.

14.15.2.2. New accumulator construction ACC^{SC}

The new accumulator scheme, denoted ACC^{SC}, is based on the accumulator introduced by *Barić and Pfitzmann* [24]. It was also jointly published in [395] (see Appendix A publication n^{o} 15).

Construction 12 : ACC^{SC}: Collision-Resistant Trapdoor Accumulator with Semi-Trusted Setup

It consists of five efficient (PPT) algorithms, s.t. $\mathsf{ACC}^{SC} = (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Dig}, \mathsf{Wit}, \mathsf{Ver})$. *While the Wit, Dig and Ver algorithms remain unchanged from Barić's and Pfitzmann's proposal, the Setup and Gen algorithms have been modified to allow for a **semi-trusted setup**. All algorithms may output $\perp$ in case of an error.*

Setup. *The algorithm Setup generates two safe primes p_1 and q_1 with bit length λ. It returns $n_1 = p_1 q_1$.*

Gen. *On input of the parameters parm, containing a modulus $n_1 = p_1 q_1$ of unknown factorisation and a security parameter λ, the algorithm outputs a multi-prime RSA-modulus $N = n_1 n_2$, where $n_2 = p_2 q_2$, where $p_2, q_2 \in \mathbb{P}$ are random safe primes with bit length λ.*

Verf. *On input of the parameters parm $= n_1$, containing a modulus $N = p_1 q_1 p_2 q_2 = n_1 n_2$ of unknown factorisation, a security parameter λ, an element y_i, an accumulator a, and a corresponding proof p_i, it checks, whether $p_i^{y_i} \pmod{N} = a$ and if $n_1 \mid N$ and $n_2 = \frac{N}{n_1} \notin \mathbb{P}$. If either checks fail, it returns 0, and 1 otherwise.*

Other algorithms: *The other algorithms work exactly like the standard collision-free RSA-accumulator, i.e., [24].*

14.15.2.3. New collision resistance property for accumulator with semi-trusted setup

The new accumulator for semi-trusted setup needs to offer the following collision resistance:

Definition 228 : Collision-Resistance for the Public Key Generator within a Semi-Trusted Setup

*An accumulator ACC with semi-trusted setup is said to be **collision-resistant for the public key generator**, if for every efficient (PPT) adversary $\mathcal{A}$, the probability that the game* $\mathsf{SemiTrustedCollisionResistancePK}_{\mathcal{A}}^{ACC}(\lambda)$ *depicted in Fig. 119 returns* 1, *is negligible (as a function of λ).*

The basic idea behind the properties description in $\mathsf{SemiTrustedCollisionResistancePK}_{\mathcal{A}}^{\mathsf{ACC}}$ is to let the adversary generate public key pk. The other part is generated by the challenger. Afterwards, the adversary has to find a collision.

```
**Experiment** $\mathsf{SemiTrustedCollisionResistancePK}_{\mathcal{A}}^{\mathsf{ACC}}(\lambda)$
    $\mathsf{parm} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
    $(\mathsf{pk}^*, p^*, m^*, a^*) \leftarrow \mathcal{A}^{\mathsf{ODig}(\cdot,\cdot)}(1^\lambda, \mathsf{parm})$
        where oracle $\mathsf{ODig}$, on input of $S_i$, $\mathsf{pk}_i$ returns:
            $(a_i, \mathsf{aux}_i) \leftarrow \mathsf{Dig}(\mathsf{pk}_i, S_i)$ (answers/queries indexed by $i$, $1 \leq i \leq k$)
            $P_i = \{(s_j, p_i) \mid p_i \leftarrow \mathsf{Proof}(\mathsf{pk}_i, \mathsf{aux}_i, s_j, S_i), s_j \in S_i\}$
            return $(a_i, P_i)$
    return 1, if:
        $\mathsf{Verf}(\mathsf{pk}^*, a^*, m^*, p^*) = 1$ and
        $\exists i, 1 \leq i \leq k : a_i = a^*$ and $m^* \notin S_i$
```

Figure 119. Collision-Resistance Experiment for Accumulator's Public Key Generator within a Semi-Trusted Setup

Definition 229 : Collision-Resistance for the Parameter Generator within a Semi-Trusted Setup

An accumulator ACC with semi-trusted setup is said to be ***collision-resistant for the parameter generator****, if for every efficient (PPT) adversary $\mathcal{A}$, the probability that the game* $\mathsf{SemiTrustedCollisionResistancePARM}_{\mathcal{A}}^{ACC}(\lambda)$ *depicted in Fig. 120 returns* 1*, is negligible (as a function of λ).*

The basic idea behind the game in Fig. 120 is to let the adversary generate only the public parameters parm, but not any public keys; they are required to be generated honestly. Afterwards, the adversary has to find a collision.

```
**Experiment** $\mathsf{SemiTrustedCollisionResistancePARM}_{\mathcal{A}}^{\mathsf{ACC}}(\lambda)$
    $(\mathsf{parm}^*, s^*) \leftarrow \mathcal{A}(1^\lambda)$
    $(\mathsf{pk}^*, p^*, m^*, a^*) \leftarrow \mathcal{A}^{\mathsf{ODig}(\cdot,\cdot),\mathsf{GetPk}()}(1^\lambda, s^*)$
        where oracle $\mathsf{ODig}$, on input of $\mathsf{pk}_i$, $S_i$:
            $(a_i, \mathsf{aux}_i) \leftarrow \mathsf{Dig}(\mathsf{pk}_i, S_i)$ (answers/queries indexed by $i$, $1 \leq i \leq k$)
            $P_i = \{(s_j, p_i) \mid p_i \leftarrow \mathsf{Proof}(\mathsf{pk}_i, \mathsf{aux}_i, s_j, S_i), s_j \in S_i\}$
            return $(a_i, P_i)$
        where oracle $\mathsf{GetPk}$ returns:
            $\mathsf{pk}_j \leftarrow \mathsf{Gen}(1^\lambda, \mathsf{parm}^*)$ (answers/queries indexed by $j$, $1 \leq j \leq k'$)
    return 1, if:
        $\mathsf{Verf}(\mathsf{pk}^*, a^*, m^*, p^*) = 1$ and
        $\exists i, 1 \leq i \leq k : a_i = a^*, m^* \notin S_i$ and $\exists j, 1 \leq j \leq k' : \mathsf{pk}^* = \mathsf{pk}_j$
```

Figure 120. Collision-Resistance Experiment for Accumulator's Parameter Generator within a Semi-Trusted Setup

14.15.2.4. Proof of collision resistance for ACC^{SC} in a semi-trusted setup

Theorem 45 : The Accumulator ACC^{SC} from Construction 12 is Collisions-Resistant with Semi-Trusted Setup.

If either the parameters parm *or the public key* pk *has been generated honestly, the sketched construction of* ACC^{SC} *given in Construction 12 is collision-resistant with semi-trusted setup.*

Proof 45

Based on the proofs given in [24], it is required to show that an adversary able to find collisions is able to find the e^{th} root of a modulus of unknown factorisation. Following the definition given in Fig. 119 and Fig. 120, one has three cases:

I) ***Malicious Semi-Trusted Third Party.*** *As* parm *is public knowledge, every party can compute $n_2 = \frac{N}{n_1}$. For this proof assume that the strong RSA-assumption [24] holds in $(\mathbb{Z}/n_1\mathbb{Z})$ and $(\mathbb{Z}/n_2\mathbb{Z})$. Moreover it is required that $\gcd(n_1, n_2) = 1$ holds. From $(\mathbb{Z}/N\mathbb{Z}) \cong (\mathbb{Z}/n_1\mathbb{Z}) \times (\mathbb{Z}/n_2\mathbb{Z})$ follows a group isomorphism φ_1. Furthermore, as the third party knows the factorisation of n_1, another group isomorphism is φ_2. It follows: $(\mathbb{Z}/N\mathbb{Z}) \cong (\mathbb{Z}/p_1\mathbb{Z}) \times (\mathbb{Z}/q_1\mathbb{Z}) \times (\mathbb{Z}/n_2\mathbb{Z})$. Assuming that $\mathcal{A}$ can calculate the e^{th} root in $(\mathbb{Z}/N\mathbb{Z})$, it implies that it can calculate the e^{th} root in $(\mathbb{Z}/n_2\mathbb{Z})$, as calculating the e^{th} root in $(\mathbb{Z}/p\mathbb{Z})$, with $p \in \mathbb{P}$ is trivial. It follows that $\mathcal{A}$ breaks the strong RSA-assumption in $(\mathbb{Z}/n_2\mathbb{Z})$. Building a simulation and an extractor is straightforward.*

II) ***Malicious Signer.*** *Similar to I).*

III) ***Outsider.*** *Outsiders have less knowledge, hence a combination of I) and II).*

□

Obviously, if the factorisation of n_1 **and** n_2 is known, one can simply compute the e$^{\text{th}}$ root in $(\mathbb{Z}/N\mathbb{Z})$. However, the above assumed that Signer and trusted third party trusted third party do not collude. All other parties can collude, as the factorisation of n_2 remains secret with overwhelming probability.

14.15.3. Construction of $merge\mathcal{RSS}$ on smart card

A practically usable *undeniable* RSS based on $merge\mathcal{RSS}$ (see Sec. 14.12) to be executable on smart cards is constructed on a standard signature scheme $\mathsf{S} := (\mathsf{S.KGen}, \mathsf{S.Sign}, \mathsf{S.Verify})$ and the newly devised accumulator with semi-trusted setup $\mathsf{ACC}^{SC} := (\mathsf{Setup}, \mathsf{Gen}, \mathsf{Dig}, \mathsf{Proof}, \mathsf{Verf})$. The details of the accumulator were presented in section 14.15.2.2.

Construction 13 : $merge\mathcal{RSS}^{SC}$: Mergeable redactable signatures scheme for smart cards

The RSS is constructed as a set of algorithms RSS $=$ (KeyGen, Sign, Verify, Redact, Link, Merge), as follows:

Key Generation: *The algorithm KeyGen generates $(sk_S, pk_S) \leftarrow$ S.KGen(1^λ). It lets parm $\leftarrow$ Setup(1^λ) and $pk_{ACC} \leftarrow$ Gen$(1^\lambda, parm)$. The algorithm returns $((pk_S, parm, pk_{ACC}), (sk_S))$.*

Signing: *Sign on input of sk_S, pk_{ACC} and a set S, it computes $(a, aux) \leftarrow$ Dig$(pk_{ACC}, (S))$. It generates $P = \{(y_i, p_i) \mid p_i \leftarrow$ Proof$(pk_{ACC}, aux, y_i, S) \mid y_i \in S\}$, and the signature $\sigma_a \leftarrow$ S.Sign(sk_S, a). The tuple (S, σ_s) is returned, where $\sigma_s = (pk_S, \sigma_a, \{(y_i, p_i) \mid y_i \in S\})$.*

Verification: *Verify on input of a signature $\sigma = (pk_S, \sigma_a, \{(y_i, p_i) \mid y_i \in S\})$, parm and a set S first verifies that σ_a verifies under pk_S using S.Verify. For each element $y_i \in S$ it tries to verify that Verf$(pk_{ACC}, a, y_i, p_i) =$ `true`. In case Verf returns `false` at least once, Verify returns `false` and `true` otherwise.*

Redaction: *Redact on input of a set S, a subset $R \subseteq S$, an accumulated value a, pk_S and a signature σ_s generated with Sign first checks that σ_s is valid using Verify. If not $\perp$ is returned. Else it returns a tuple (S', σ_s'), where $\sigma_s' = (pk_S, \sigma_a, \{(y_i, p_i) \mid y_i \in S'\})$ and $S' = S \setminus R$.*

Linking: *Link on input of two sets S and T, signatures σ_S and σ_T, the accumulated values a_S and a_T and pk_S first tries to verify that both σ_S and σ_T are valid signatures using Verify, else $\perp$ is returned. It returns 1, if $a_S = a_T$, 0 otherwise.*

Merging: *Merge on input of two sets S and T, signatures σ_S and σ_T, the accumulated values a_S and a_T and pk_S the algorithm first checks that $\mathsf{Link}(\mathsf{pk}_S, S, \sigma_S, T, \sigma_T, a_S, a_T) = 1$, else $\perp$ is returned. It returns $(S \cup T, \sigma_U)$, where $\sigma_U = (\mathsf{pk}_S, \sigma_a, \{(y_i, p_i) \mid y_i \in S \cup T\, p_i \in P\})$*

14.15.4. Security properties of the smart card version of $merge\mathcal{RSS}$

The proofs for the security properties in Sec. 14.12.4 were based on the construction $merge\mathcal{RSS}$ as given in Sec. 14.12.2 on the assumption that the underlying accumulator is strongly collision-resistant. This section proofs the properties for he accumulator with semi-trusted setup that was designed to be practically on a smart card. Note, Construction 13's security proofs were also presented in the joint work published at WISTP 2013 [395] (see Appendix A publication n^o 15).

Further note, the privacy is already implied by $merge\mathcal{RSS}$ being transparent.

Unforgeability: Given a set of signed messages S and the accumulated value a for that set, one way to break the scheme's unforgeability would be for an adversary to find a set of messages $S' \nsubseteq S$, that produces an accumulated value a' where $a' = a$. This contradicts the collision resistance of the underlying standard signature scheme.
Another way would be to find a set $S' \nsubseteq S$ that produces an accumulated value a', where $a' \neq a$, but $\mathsf{S.Sign}(\mathsf{sk}_{adversary}, a') = \mathsf{S.Sign}(\mathsf{sk}_{\mathrm{sig}}, a)$. This again would contradict the unforgeability of the underlying standard signature scheme.
The scheme is therefore unforgeable, if the cryptographic accumulator is collision-resistant and the underlying standard signature is unforgeable.

Transparency: To break the scheme's transparency an adversary would have to extract the members of the original set, accumulated into the accumulated value a, from a. This contradicts the accumulator's one-way property. The scheme therefore provides transparency.

Committing: The scheme is committing if a cryptographic accumulator without trusted setup is used. To break this property the signer would have to be able to calculate new witnesses for elements m', that are not part of the set the signer originally committed to. This, however, would contradict the definition of security without trusted setup (see section 14.12.1.1).

Linkability: As signatures over subsets derived from the same set S share the common accumulated value a and thereby the same standard signature σ_S over a, one way for an adversary to break linkability is to forge σ_S by breaking the unforgeability of the underlying standard signature scheme.
Another way would be to find a set T that is not a subset of S and that shares the same accumulated value a with S. This, however, would contradict the collision-resistance of the cryptographic accumulator. Therefore the $merge\mathcal{RSS}$ provides linkability, if the standard signature scheme is unforgeable and the cryptographic accumulator provides collision-resistance.

Mergeability: To break the mergeability of the scheme the adversary would have to find two sets, which are linkable to a set S, but not mergeable. In other words for the accumulated value a for S, it can find an element $m' \notin S$ and a witness π for m' with respect to a, for which the Verify algorithm outputs `valid`. Analogously to the proof sketch for linkability this would either break the unforgeability of the underlying standard signature scheme or the collision-resistance of the cryptographic accumulator. The scheme is therefore mergeable.

14.15.5. Smart card implementation overview

The KeyGen algorithm and the Sign algorithm of the scheme needed to reside on card, as they require the knowledge of private information. This scheme involves the smart card for the algorithms Setup and Sign as illustrated in Figures 121 and 122. The implementation of the required cryptographic accumulator is using the construction with semi-trusted setup from Sec. 14.15.2.2. The semi-trusted setup guarantees that the scheme is committing, as long as the party involved in the Gen algorithm does not collaborate with the party involved in the Setup algorithm. The accumulator additionally requires a function $\mathcal{PH}$ that maps its input to a random prime number.

Prime hash $\mathcal{PH}$. This function implemented basically looks for the closest prime number greater than a cryptographic hash digest of n and returns the prime. It is defined as follows:

1. Calculate $h' = \mathcal{H}(n)$, where $\mathcal{H}$ is a cryptographic hash-function.
2. Let $h = \begin{cases} h'+1 & \text{if } h' \text{ is even} \\ h' & \text{otherwise} \end{cases}$.
3. While h is not prime, let $h := h + 2$.
4. Output h.

According to *Dusart* [158] this is guaranteed to produce a prime number from the input of any number $n > 396738$ after at most $\frac{(1+\frac{1}{(25 \ln 2n)})n}{2} - 1$ repetitions of step 3.
As the output of the cryptographic hash-function should be sufficiently large, the implementation uses the SHA-512 algorithm to implement $\mathcal{H}$.

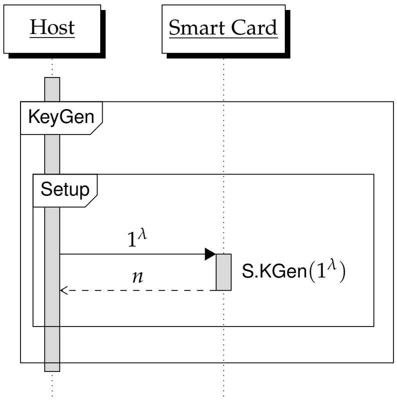


Figure 121. *merge$\mathcal{RSS}$*: Interactions between host and smart card during KeyGen algorithm; as jointly published in *Pöhls, Peters, Samelin, Posegga, and de Meer* [395]

KeyGen. During Setup the smart card is required because it generates the Signer's secret key obtain the blinding primes of the modulus as needed by Setup (see Sec. 14.15.2.1). Part of the KeyGen algorithm is assumed to be executed by a trusted third party. The sub-algorithm Setup then provides a modulus n_2. The smart card implements only the sub-algorithm Gen from the KeyGen algorithm on the smart card. When Gen is executed on the smart card it itself executes S.KGen of its RSA implementation, takes the modulus from the result and discards the additional parameters. The modulus is stored on card. The card additionally generates the key pair $\mathsf{sk_S}, \mathsf{pk_S}$, again using S.KGen, and stores it. To compute these primes on card, the implementation generates standard RSA parameters (N, e, d) with N being of 2048 Bit length, but store only N on card and discard the exponents. On the host system this modulus is multiplied with that obtained from the trusted third party to form the modulus used by ACC.

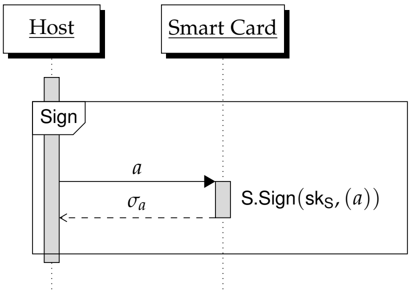


Figure 122. *merge$\mathcal{RSS}$*: Interactions between host and smart card during the Sign algorithm; as jointly published in *Pöhls, Peters, Samelin, Posegga, and de Meer* [395]

Sign. During the Sign algorithm the accumulated value a is transmitted to the card. The card signs the value, using a build in standard RSA signature and the secret key $\mathsf{sk_S}$ protected inside the smart card, and returns the resulting signature σ_a to the host.

14.15.6. Performance evaluation of $\mathit{merge}\mathcal{RSS}$ on a smart card

Tab. 25 shows the performance of $\mathit{merge}\mathcal{RSS}$ on the smart card in comparison to the speed obtained by the sanitizable signature scheme that was discussed in Sec. 13.10. The key lengths for the underlying digital signature scheme of the smart card was fixed to 2048 bit. In the implementation this key length controls also the length of the moduli used in the underlying accumulator scheme by both the host and the smart card. As the accumulator described in Sec. 14.15.2.2 is built upon the multiplication of multiple primes, the length of the modulus used in the accumulator scheme is twice the specified key length.

	Sign**			Redact**			Verify		
ℓ	10	25	50	10	25	50	10	25	50
$\mathit{merge}\mathcal{RSS}$	11.16	59.97	221.97	1.42	3.17	6.32	1.32	3.12	6.12
$\mathit{pubacc}\mathcal{SSS}$	3.12	7.16	13.24	2.60	6.65	12.74	0.016	0.039	0.084

	Judge			Detect/ Proof		
ℓ	10	25	50	10	25	50
$\mathit{merge}\mathcal{RSS}$	-*	-*	-*	-*	-*	-*
$\mathit{pubacc}\mathcal{SSS}$	0.043	0.051	0.060	0.001	0.001	0.002

*algorithm not defined by scheme **involves smart card operations

Table 25. Performance of $\mathit{merge}\mathcal{RSS}$ and $\mathit{pubacc}\mathcal{SSS}$ (for comparison) prototype on smart card; median runtime in seconds; as published in *Pöhls, Peters, Samelin, Posegga, and de Meer* [395]

15 — Separating secure RSS from secure SSS

Overview of Chapter 15

This thesis considers SSS and RSS as tools with separate functionality. Already *Ateniese et al.* noted that “[...] there are other fundamental differences between sanitizable signatures and redactable signatures.” [12], but they provided no further formal separation [12].

Chapter 15 separates RSS from SSS[858]. It shows rigorously that an RSS is not trivially a special case[859] of SSS. The findings are:

- no black-box transformation can transform an RSS into an SSS, and
- a black-box transformation of an SSS even with additionally strengthened security only yields an RSS with weaker security properties than the state of the art. [133]

These findings are the reason for the split focus on RSS and SSS of this thesis (see Sec. 1.4). Tab. 26 shows the new properties from this thesis that facilitated the separation. Some properties have already been raised before or are discussed later; the table contains a reference to the section where each relevant property is introduced.

Property	**Scheme’s name**	**Section**	**Publication(s)**
Weak immutability for SSS (15.1.2)	–	15.1.2	[133]
Strengthened privacy for RSS (15.1.3)	–	15.1.3	[133]
Weak privacy for RSS (15.1.4)	–	15.1.4	[133]
Accountability, both transparent and publicly non-interactive, on the scope of individual or groups of block(s) (6.5, 13.2, 15.1.6)	$\mathit{blockacc}\mathcal{SSS}$	13.6	[132, 133]
Strengthened unlinkability for SSS (13.3)	$\mathit{unlinkable}\mathcal{SSS}$	13.8	[69]

Table 26. Overview of proposed SSS and RSS properties required to separate secure RSS from secure SSS

This separation concludes Part III. In previous chapters of Part III RSS and SSS were improved separately (see Chapter 13 for SSS and Chapter 14 for RSS) to fulfil the requirements defined in Parts I and II.

[858] The chapter’s results on the separation of RSS and SSS have in part been published at the 2014 ESSoS conference [133] as joint work with *H. de Meer*, *J. Posegga*, and *K. Samelin* (see Appendix A publication nº 19).

[859] This misconception might have been arising after non-careful study of existing works: *Yuen et al.* [508] state “Type 4 [offering SSS functionality where blocks can be sanitized into arbitrary messages] can be converted to type 2 [offering RSS functionality where a redacted block is represented by a special character ∅ visible to Verifiers] by using a special character ∅.” [508] without giving an explicit analysis of the impact on the property of privacy as defined in previous papers. Instead the authors concentrate on the property of transparency. *Yum et al.* [510] state: “Informally, we will use the term of sanitizable signature to refer to any signature scheme for the digital document sanitizing problem.” [510].

15.1. Separation of security models

The following results on the separation of RSS and SSS have in part been published at the 2014 ESSoS conference [133] as joint work with *H. de Meer*, *J. Posegga*, and *K. Samelin* (see Appendix A publication n^{o} 19).

Apart from the following works that do not separate but combine the two strands, the author of this thesis is not aware of any work considering relations between them: Combinations of RSS and SSS can be found in the literature, e.g., [222, 265, 266, 508]. The work of *Yuen et al.* [508] presents a model that jointly expresses RSS and SSS and shows some conversions between the levels/types they defined for certain properties, e.g. four types of how a sanitized message looks. However, those schemes do not preserve the property of privacy with a strength equal or better than standard privacy based on *Brzuska et al.*; the analysis is given in Shortcoming 1[860].

The following statements assume that for RSS and SSS the usual correctness properties hold. In particular, all genuinely signed or sanitized messages are accepted, while every genuinely created proof π by the Signer leads the judge to decide in favour of the Signer. Those are given in full later in Sec. 11.11 and Sec. 11.4. It is also required by every SSS that ADM is always correctly recoverable from any valid message-signature pair (m, σ). This accounts for the work done by *Gong et al.* [216].

Following *Brzuska et al.* [64] and the motivational examples an SSS must at least be unforgeable, immutable, accountable and private to be meaningful. The thesis agrees and assumes henceforth that all used SSSs fulfil these four fundamental security requirements; if these requirements are not met, the construction is not considered a secure SSS and the results of this separation are not directly applicable. In the same way an RSS must be unforgeable and (weakly) private to be meaningful for the application domains of RSS given so far in this thesis, as well as for the applications presented by *Brzuska et al.* [66]. *Weak privacy* is a new adjusted property used in this separation. Regarding properties, this section is also based on the clear separation of accountability into PUB and INT, as described by the 3rd aspect of the extended integrity definition from Sec. 6.4. In terms of algorithms on block-level, the difference between INT and PUB was formalised as the new property: *block-level non-interactive public accountability* for SSS (see Definition 190 in Sec. 13.2.2). The separation will require a special weak form of this property. The latter is presented in more depth in this section (see Sec. 15.1.6). In general the block-level non-interactive public accountability (PUB) is of importance for probative value: The impact of PUB has been formulated as Requirement 4[861] and the missing fine-grained scope of individual blocks has been identified as Shortcoming 3[862].

15.1.1. The property of unforgeability remains unchanged

There is no need to adjust the definition for the property of unforgeability in any manner. It was already harmonised in this thesis and their formal definitions based on existing work can be found in Definition 152 for SSS and in Definition 172 for RSS.

15.1.2. The property of weak immutability for SSS is introduced

The immutability for SSS as given in Definition 154 needs to be further detailed for separation. The idea behind immutability is that an adversary generating the Sanitizer key must only be able to sanitize admissible blocks. Hence, immutability is the unforgeability requirement for the Sanitizer.

In the stronger version of immutability — based off the works of *Brzuska et al.* [64] and *Gong et al.* [216] — the adversary generates and supplies the Sanitizer's public key to the oracle; in a weaker immutability version, an adversary knowing, but not generating the sanitizer key must only be able to sanitize admissible blocks. Hence, compared to the game $\mathsf{Immutability}_{\mathcal{A}}^{\mathsf{SSS}}$ in Fig. 48 $\mathsf{pk}_{\mathsf{san}}$ is fixed in the game $\mathsf{WImmutability}_{\mathcal{A}}^{\mathsf{SSS}}$ and tracking of $\mathsf{pk}_{\mathsf{san}}$ in $\mathcal{Q}$ can be omitted. Weak immutability gets defined as follows:

[860] Shortcoming 1: Not all schemes achieve sophisticated levels of privacy (see Sec. 12.1).

[861] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[862] Shortcoming 3: No, or no explicit public form of accountability (see Sec. 12.3).

Definition 230 : SSS Weak Immutability

> *A sanitizable signature scheme* SSS *is weakly immutable, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment* $\mathsf{WImmutability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ *given in Fig. 123 returns* 1 *is negligible (as a function of* λ*).*

```
Experiment $\mathsf{WImmutability}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\text{sig}}, \mathsf{sk}_{\text{sig}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $(\mathsf{pk}_{\text{san}}, \mathsf{sk}_{\text{san}}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $\mathcal{Q} \leftarrow \emptyset$
  $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda, \cdot, \mathsf{sk}_{\text{sig}}, \mathsf{pk}_{\text{san}}, \cdot), \mathsf{Proof}(1^\lambda, \mathsf{sk}_{\text{sig}}, \cdots, \mathsf{pk}_{\text{san}}, \cdot)}(1^\lambda, \mathsf{pk}_{\text{sig}}, \mathsf{pk}_{\text{san}}, \mathsf{sk}_{\text{san}})$
  let $(m_i, \mathsf{ADM}_i)$ and $\sigma_i$ for $i = 1, 2, \ldots q$ be queries/answers to/by Sign
  For each query to Sign, do:
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(\mathsf{ADM}_i, m_i') \mid m_i' \in \{\mathsf{MOD}(m_i)\}^{\mathsf{ADM}_i}\}$ //$\mathcal{Q}$ records all admissible messages
  return 1, if:
    $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\text{sig}}, \mathsf{pk}_{\text{san}}) = \texttt{true}$ and
    $(\mathsf{ADM}^*, m^*) \notin \mathcal{Q}$
```

Figure 123. Weak Immutability Experiment for SSS

Interestingly, weak immutability is enough for a construction of an SSS to stay unforgeable, while for an RSS used in the normal way, this definition is obviously not suitable at all due to accountability reasons.

15.1.3. The property of strengthened privacy for SSS is introduced

Standard privacy has been defined and harmonised in this thesis and a formal definition based on existing work can be found in Definition 156 for SSS. In the aforementioned standard privacy definition for SSS the challenger generates the Sanitizer's key pair. Namely, the secret sanitization key sk_{san} is not given to the adversary in the experiment $\mathsf{Privacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ given in Fig. 51 on page 265 from Definition 156. Thus, standard privacy only considers outsiders — party without control or access to secret keys — as adversarial. To strengthen privacy one can require that even insiders, i.e., adversarial Sanitizers, are not able to win the game. Still sk_{san} is *not* generated by the adversary, but it might be known to it. For a strengthened definition of privacy, the basic idea remains the same: No one should be able to gain any knowledge about sanitized parts without having access to them, with one exception: The adversary is given the secret key sk_{san} of the Sanitizer. This notion of *strengthened privacy* extends the definition of standard privacy from Definition 156 to also account for parties knowing the secret sanitizer key sk_{san}. In a sense, this definition captures some form of *forward security*. Forward in the sense that "[f]orward-secure digital signatures [...] ensure that past signatures remain secure even if the current secret key is leaked." [262]. In the setting of MSS this would mean that even if sk_{san} is leaked, previously sanitizations can not be reverted, i.e. privacy is maintained. Of course, with the knowledge of sk_{san} the adversary can sanitize messages even further. Examples for such SSSs with strengthened privacy are the schemes introduced by *Brzuska et al.* [65] and the constructions $\mathit{pubaccSSS}$ (see Sec. 13.5.1) and $\mathit{unlinkableSSS}$ (Sec. 13.9.2) as these three schemes are perfectly private[863].

It gets defined as follows:

Definition 231 : Strengthened Privacy for SSS

> *An* SSS *offers* ***strengthened privacy****, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the experiment* $\mathsf{StrengthenedPrivacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$ *given in Fig. 124 returns* 1 *is negligibly close to* $\frac{1}{2}$ *(as a function of* λ*).*

The game in Fig. 124 for strengthened privacy only differs from the standard privacy experiment (see Definition 156 and Fig. 51) in one facet: The secret sanitizer key sk_{san} is given to the adversary $\mathcal{A}$ - as a third parameter.

[863] Perfect here means the property holds in an information-theoretical sense.

In the strengthened privacy game (Fig. 124) the adversary can use signature and proof oracles to generate and verify signed messages, as in the standard privacy definition. The difference — comparison of the experiments Fig. 60 and Fig. 124 — is that the adversary does no longer need an oracle for sanitizing, as it can sanitize messages using its knowledge of $\mathsf{sk}_{\mathrm{san}}$.

To breach privacy, the adversary has to decide which of two input messages was randomly chosen and used by the oracle to produce the signed outcome. The adversary can adaptively give two crafted input messages and modification instructions to the LoRSanit oracle. Note, the adversary is still not generating any signatures himself; all signatures are generated honestly, as in the standard privacy definition. Of course, trivial ways to win the game are excluded. This is described with the five abort conditions: First it is checked if both modifications of the supplied messages would result in the same sanitized message ($\mathsf{MOD}_0(m_0) = \mathsf{MOD}_1(m_1)$). The second and third condition exclude cases where the outputted sanitized message would differ due to errors induced due to specially crafted input where the allowed modifications (ADM) on the messages are non-compliant with the requested modification instructions (MOD). The fourth is catering for the results of the recent work of *Krenn et al.* [294]: It ensures that the supplied policies ($\mathsf{ADM}_{\{0,1\}}$) before and after modification match the respective message. The final abort condition ($\mathsf{MOD}_0(\mathsf{ADM}_0) \neq \mathsf{MOD}_1(\mathsf{ADM}_1)$) means that — if a modification to ADM is allowed by the scheme — the ADM' of the resulting sanitized message m' shall be the same regardless what input message was used for the sanitization. This was not found in the original game in [66] for standard privacy. It is added to cater for schemes that would offer consecutive sanitization control (see Sec. 14.5.1.1 for redactions), which is then modelled as an authorized modification of ADM.

```
**Experiment** $\mathsf{StrengthenedPrivacy}_{\mathcal{A}}^{\mathsf{SSS}}(\lambda)$
  $(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}) \leftarrow \mathsf{KGen}_{\mathrm{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{KGen}_{\mathrm{san}}(1^\lambda)$
  $b \overset{\$}{\leftarrow} \{0,1\}$
  $a \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\mathsf{pk}_{\mathrm{san}},\cdot),\mathsf{Proof}(1^\lambda,\mathsf{sk}_{\mathrm{sig}},\cdots,\mathsf{pk}_{\mathrm{san}}),\mathsf{LoRSanit}(1^\lambda,\cdots,\mathsf{sk}_{\mathrm{sig}},\mathsf{sk}_{\mathrm{san}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}})$
    where oracle LoRSanit on input of $m_0, \mathsf{MOD}_0, m_1, \mathsf{MOD}_1, \mathsf{ADM}$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$ // aborts if resulting $m'$ is not the same
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}$, return $\perp$
      if $\mathsf{ADM} \not\subseteq m_0$ or $\mathsf{ADM} \not\subseteq m_1$ or
        $\mathsf{MOD}_0(\mathsf{ADM}) \not\subseteq \mathsf{MOD}_0(m_0)$ or $\mathsf{MOD}_1(\mathsf{ADM}) \not\subseteq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}) \neq \mathsf{MOD}_1(\mathsf{ADM}_1)$, return $\perp$
      let $(m_i, \sigma_i) \leftarrow \mathsf{Sign}(1^\lambda, m_b, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{ADM})$
      return $(m', \sigma') \leftarrow \mathsf{Sanit}(1^\lambda, m_i, \mathsf{MOD}_b, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$
  return 1, if $a = b$
```

Figure 124. Strengthened Privacy Experiment for SSSs

15.1.4. The property of weak privacy for RSS is introduced

Standard privacy has been defined and harmonised in this thesis and a formal definition based on existing work can be found in Definition 174 for RSS. In a weakly private RSS, a third party can derive which blocks of a message have been redacted without gathering more information, as redacted blocks are replaced with ■, which is visible. The basic idea is that the oracle either signs and sanitizes the first message (m_0) or the second (m_1). As before, the resulting redacted message m' must be the same for both inputs, with one additional exception: The length of both inputs must be the same, while ■ is considered part of the message. For strong privacy, this constraint is not required.

Definition 232 : RSS Weak Privacy

> *An RSS is* ***weakly*** *private, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the game depicted in Fig. 125 returns* 1*, is negligibly close to $\frac{1}{2}$ (as a function of λ).*

Weakly private schemes, following this definition, are, e.g. [222, 295]. In their schemes, the adversary is able to pinpoint the indices of the redacted blocks, as ■ is visible.

```
Experiment $\mathsf{WPrivacy}_{\mathsf{RSS},\mathcal{A}}(\lambda)$
  $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $b \leftarrow \{0,1\}$
  $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{LoRRedact}(1^\lambda,\cdots,\mathsf{sk},b)}(1^\lambda,\mathsf{pk})$
    where oracle LoRRedact for input $m_0, m_1, \mathsf{MOD}_0, \mathsf{MOD}_1$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}_0$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}_1$, return $\perp$
      if $\mathsf{ADM}_0 \not\subseteq m_0$ or $\mathsf{ADM}_1 \not\subseteq m_1$
        or $\mathsf{MOD}_0(\mathsf{ADM}_0) \not\subseteq \mathsf{MOD}_0(m_0)$ or $\mathsf{MOD}_1(\mathsf{ADM}_1) \not\subseteq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}_0) \neq \mathsf{MOD}_1(\mathsf{ADM}_1)$, return $\perp$
      // Note: redacted blocks are considered part of the message and denoted with the symbol '■':
      if $|m_0| \neq |m_1|$, return $\perp$
      check if $\forall 1 \leq i \leq |m_0|$ and $m_0[i] = \blacksquare : m_1[i] = \blacksquare$, otherwise return $\perp$
      $(m,\sigma) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, m_b)$
      return $(m',\sigma') \leftarrow \mathsf{Redact}(1^\lambda, \mathsf{pk}, m, \sigma, \mathsf{MOD}_b)$.
  return 1, if $b = d$
```

Figure 125. Weak Privacy Experiment for RSS

15.1.5. The property of strong privacy for RSS is introduced

Standard privacy has been defined and harmonised in this thesis and a formal definition based on existing work can be found in Definition 174 for RSS. In the stronger version of the definition, redacted blocks are *not* considered part of the message. This would allow the attacker to input messages of different length which are differently redacted to the same message. For example, $m_0 = A|B|C$ with $\mathsf{MOD}_0 = 1,2$ and $m_1 = A|D$ with $\mathsf{MOD}_1 = 1$ would need to result in a redacted message $m' = A$ from which the attacker can not deduce its origin. To make this point clear for the discussion on the separation of RSS and SSS this length-hiding[864] redaction is denoted as *strong privacy* and defined as follows:

Definition 233 : RSS Strong Privacy

> *An RSS is strongly private, if for any efficient (PPT) adversary $\mathcal{A}$ the probability that the game depicted in Fig. 126 returns* 1*, is negligibly close to $\frac{1}{2}$ (as a function of λ).*

```
Experiment $\mathsf{SPrivacy}_{\mathcal{A}}^{\mathsf{RSS}}(\lambda)$
  $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  $b \leftarrow \{0,1\}$
  $d \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{LoRRedact}(1^\lambda,\cdots,\mathsf{sk},b)}(1^\lambda,\mathsf{pk})$
    where oracle LoRRedact for input $m_0, m_1, \mathsf{MOD}_0, \mathsf{MOD}_1$:
      if $\mathsf{MOD}_0(m_0) \neq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{MOD}_0 \not\subseteq \mathsf{ADM}_0$, return $\perp$
      if $\mathsf{MOD}_1 \not\subseteq \mathsf{ADM}_1$, return $\perp$
      if $\mathsf{ADM}_0 \not\subseteq m_0$ or $\mathsf{ADM}_1 \not\subseteq m_1$
        or $\mathsf{MOD}_0(\mathsf{ADM}_0) \not\subseteq \mathsf{MOD}_0(m_0)$ or $\mathsf{MOD}_1(\mathsf{ADM}_1) \not\subseteq \mathsf{MOD}_1(m_1)$, return $\perp$
      if $\mathsf{MOD}_0(\mathsf{ADM}_0) \neq \mathsf{MOD}_1(\mathsf{ADM}_1)$, return $\perp$
      // Note: redacted blocks are not considered part of the message
      $(m,\sigma) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, m_b)$
      return $(m',\sigma') \leftarrow \mathsf{Redact}(1^\lambda, \mathsf{pk}, m, \sigma, \mathsf{MOD}_b)$.
  return 1, if $b = d$
```

Figure 126. Strong Privacy Experiment for RSS

[864] In the joint work [392] such length-hiding properties have been proposed and schemes constructed that thus achieve strong privacy (see Appendix A publication n° 24).

15.1.6. The property of weak block-level non-interactive public accountability for SSS is introduced

The basic idea is that an adversary, i.e., the Sanitizer, has to be able to make the blkJudge algorithm accuse the Signer, even if the Signer did not sign that specific block. Note, in the following definition the Signer is **not** considered adversarial — contrary to the strengthened unlinkability notion given in Sec. 13.3 — . An example for a weakly block-level non-interactive publicly accountable SSS is the scheme introduced in this thesis as *unlinkable$\mathcal{SSS}$* (see Sections 13.8 and 13.9). Note, in the following game pk_{san} is fixed for the oracles. For SSSs the adversary is given access to the usual signing and proof oracles [64] (see Sec. 11.6.1 for the harmonised definition).

This property is defined to see if an SSS which offers accountability can be used to emulate an RSS. Note, accountability, as defined for SSSs in [64], has not been defined for standard RSS; Redact is a public algorithm. As no secret Sanitizer key(s) are required for redactions, a dedicated party can not be held accountable. This inconsistency can be circumvented as follows: Utilise a 'standard' secure SSS and let the Signer generate the Sanitizer key sk_{san}, then attach it to the public key of the Signer. This also explains why pk_{san} is fixed in this security model. If any alteration without sk_{san} is possible, the underlying SSS would obviously be forgeable. As the underlying SSS is secure this case is omitted. Hence, the formerly secret sk_{san} becomes public knowledge and can be used by every party. Further note, the adversary only knows sk_{sig}, but cannot generate it. At first sight the above restrictions might feel unnatural. However, they allow to stay consistent with the existing model of SSSs as formalised in [64]. Moreover, the Signer is generally *not* considered an adversarial entity in RSSs [66]. If other notions or adversary models are used, the result of a separation may obviously differ. In Sec. 15.3 this thesis shows that any SSS which only achieves standard privacy, is not enough to construct a weakly private RSS and additional impossibility results.

Definition 234 : SSS Weak Block-level Non-Interactive Public Accountability

*A sanitizable signature scheme SSS is **weakly non-interactive publicly accountable**, if Proof $= \perp$, and if for any efficient algorithm $\mathcal{A}$ the probability that the experiment WBlockPubAcc$_{\text{SSS},\mathcal{A}}(\lambda)$ given in Fig. 127 returns 1 is negligible (as a function of λ).*

```
Experiment WBlockPubAcc$_{\mathsf{SSS},\mathcal{A}}(\lambda)$
  $(\mathsf{pk}_{\text{sig}}, \mathsf{sk}_{\text{sig}}) \leftarrow \mathsf{KGen}_{\text{sig}}(1^\lambda)$
  $(\mathsf{pk}_{\text{san}}, \mathsf{sk}_{\text{san}}) \leftarrow \mathsf{KGen}_{\text{san}}(1^\lambda)$
  $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(1^\lambda,\cdot,\mathsf{sk}_{\text{sig}},\mathsf{pk}_{\text{san}},\cdot),\mathsf{Proof}(1^\lambda,\cdot,\mathsf{sk}_{\text{sig}},\cdots,\mathsf{pk}_{\text{san}})}(1^\lambda, \mathsf{pk}_{\text{san}}, \mathsf{sk}_{\text{san}}, \mathsf{pk}_{\text{sig}})$
    let $(m_i, \mathsf{ADM}_i)$ and $(m_i, \sigma_i)$ for $i = 1, \ldots, k$ be queries/answers to/by Sign
    return 1, if
      $\mathsf{Verify}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\text{sig}}, \mathsf{pk}_{\text{san}}) = \mathtt{true}$ and
      $\exists q$, s.t. $\mathsf{blkJudge}(1^\lambda, m^*, \sigma^*, \mathsf{pk}_{\text{sig}}, \mathsf{pk}_{\text{san}}, q) = \mathtt{Sig}$ and
      for all $i = 1, \ldots, k : (m^*[q], \sigma^*) \neq (m_i[q], \sigma_i)$.
  return 0
```

Figure 127. Weak Block-level Non-Interactive Public Accountability Experiment for SSS

15.1.7. Implications between new properties and resulting separation of SSS and RSS

In the following the first theorems of separation:

Theorem 46 : Weak Privacy $\not\Rightarrow$ Privacy

There exists an RSS which is only weakly private.

Proof 46

See [222, 265, 266, 314] for examples. □

Theorem 47 : Immutability $\Rightarrow$ Weak Immutability

Every SSS which is immutable, is also weakly immutable.

Proof 47

Trivially implied: $\mathcal{A}$ generates the sanitizer key pair honestly. □

Theorem 48 : Standard Privacy $\not\Rightarrow$ Strong Privacy

There exists an SSS which is private, but not strongly private.

Proof 48

To prove Theorem 48 an arbitrary existing strongly private SSS is altered. Let SSS = $(\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge})$ be an arbitrary private SSS. The scheme is altered as follows:

- $\mathsf{KGen}'_{sig} := \mathsf{KGen}_{sig}$
- $\mathsf{KGen}'_{san} := \mathsf{KGen}_{san}$*, while an additional key pair for a IND-CCA2-secure encryption scheme ENC is generated.*
- Sign' *is the same as Sign, but it appends the encryption e of a hash-value of original message to the final signature, i.e.,* $\sigma' = (\sigma, e)$*, where* $e \leftarrow \mathsf{ENC}(\mathsf{pk}_{san}, \mathcal{H}(m))$ *and* $\mathcal{H}$ *some cryptographic hash-function.*
- Sanit' *is the same as Sanit, while it first removes the encrypted hash-value from the signature, appending it to the resulting signature.*
- Verify', Proof' *and* Judge' *work the same as their original counterparts, but removing the trailing e from the signature before proceeding.*

A Sanitizer holding the corresponding secret key for ENC, can distinguish between messages generated by the Signer and the Sanitizer using the information contained in the signature σ*. Without* sk_{san}*, this information remains hidden due to the IND-CCA2 encryption.* □

15.2. Generic transformation of an SSS into an RSS

This section presents the generic transform. In particular, it provides a generic algorithm which transforms any weakly immutable, strongly private, and weakly blockwise non-interactive publicly accountable SSS into an unforgeable and weakly private RSS.

The basic idea of the transform is that every party, including the Signer, is allowed to alter *all* given blocks. By definition, a part containing only the symbol ■ is treated as a redacted block. The verification procedure accepts sanitized blocks, if the altered blocks are ■. Hence, redaction is altering a given block to a special symbol.

Initially, the message space of SSS has been defined to only contain strings $m[i] \in \{0,1\}^*$ (see Definition 1 on page 21); to capture the altered blocks, this section re-defines it to codify the additional symbol ■ as follows:

$$\begin{aligned}&\blacksquare := \emptyset\\ &m[i] \leftarrow 0, \text{if } m[i] = \emptyset \quad \text{and}\\ &m[i] \leftarrow m[i] + 1 \text{ else.}\end{aligned}$$

In the above $\emptyset$ expresses the empty string. With this adjustment, ■ is codified explicitly within the message space $\{0,1\}^*$ as defined initially.

Another notion defined above is weak blockwise non-public interactive accountability (see Sec. 15.1.6); for it to hold the modifications to *each* block need to be detectable to allow for a meaningful result. As SSS allows for arbitrary alterations and as ■ is still visible, the resulting scheme is only weakly private, because statements about m can be made. This contradicts the given definition of strong privacy for RSSs. Moreover, as an RSS allows every party to redact blocks, it is obvious that $\mathsf{sk}_{\mathsf{san}}$ must be known to every party, including the Signer. Therefore, a strongly private SSS is required to achieve the given definition of weak privacy for the RSS. The proof is presented in Sec. 15.3.

Construction 14 : Weakly private RSS from strongly private SSS

Let SSS $:= (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge}, \mathsf{blkJudge})$ be a secure SSS. Define RSS $:= (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Redact})$ as follows:

Key Generation: *Algorithm KeyGen generates on input of the security parameter λ, a key pair $(pk_{sig}, sk_{sig}) \leftarrow \mathsf{SSS.KGen}_{sig}(1^\lambda)$ of the SSS, and also a Sanitizer key pair $(pk_{san}, sk_{san}) \leftarrow \mathsf{SSS.KGen}_{san}(1^\lambda)$. It returns $(sk, pk) = (sk_{sig}, (sk_{san}, pk_{san}, pk_{sig}))$*

Signing: *Algorithm RSS.Sign on input $m \in \{0,1\}^*, sk, pk$, sets $\mathsf{ADM} = (1, \ldots, \ell)$ and computes $\sigma \leftarrow \mathsf{SSS.Sign}(1^\lambda, m, sk_{sig}, pk_{san}, \mathsf{ADM})$. It outputs: (m, σ)*

Redacting: *Algorithm RSS.Redact on input message m, modification instructions MOD, a signature σ, keys $pk = (sk_{san}, pk_{san}, pk_{sig})$, first checks if σ is a valid signature for m under the given public keys using RSS.Verify. If not, it stops outputting $\perp$. Afterwards, it sets $\mathsf{MOD}' = \{(i, \blacksquare) \mid i \in \mathsf{MOD}\}$. In particular, it generates a modification description for the SSS which sets block with index $i \in \mathsf{MOD}$ to $\blacksquare$. Finally, it outputs $(m', \sigma') \leftarrow \mathsf{SSS.Sanit}(1^\lambda, m, \mathsf{MOD}', \sigma, pk_{sig}, sk_{san})$.*

Verification: *Algorithm RSS.Verify on input a message $m \in \{0,1\}^*$, a signature σ and pk first checks that $\mathsf{ADM} = (1, \ldots, \ell)$ and that σ is a valid signature for m under the given public keys using SSS.Verify. If not, it returns `false`. Afterwards, for each i for which $\mathsf{SSS.blkJudge}(1^\lambda, m, \sigma_s, pk_{sig}, pk_{san}, i)$ returns `San`, it checks that $m[i] = \blacksquare$. If not, it returns `false`. Else, it returns `true`. One may also check, if sk_{san} is correct and that all $m[i]$ are sanitizable, if required.*

Theorem 49 : RSS from Construction 14 is Secure, but only Weakly Private.

If the utilised SSS is weakly blockwise non-interactive publicly accountable, weakly immutable and strongly private, the resulting RSS is weakly private, but not strongly, and unforgeable.

The proof of Theorem 49 must show that the resulting RSS is unforgeable and weakly private, but not strongly. Each property is proven individually as follows:

Proof 49

I) ***Unforgeability.*** *Let $\mathcal{A}$ be an algorithm breaking the unforgeability of the resulting RSS. The goal is to construct an algorithm $\mathcal{B}$ which breaks the weak blockwise non-interactive public accountability of the utilised SSS. To do so, $\mathcal{B}$ simulates $\mathcal{A}$'s environment in the following way:*

1. *$\mathcal{B}$ receives $pk_{san}, sk_{san}, pk_{sig}$ and forwards them to $\mathcal{A}$*
2. *$\mathcal{B}$ forwards every query to its own signing oracle*
3. *Eventually, $\mathcal{A}$ outputs a tuple (m^*, σ^*)*
4. *If (m^*, σ^*) does not verify or is trivial, abort*
5. *$\mathcal{B}$ outputs (m^*, σ^*)*

m cannot be derived from any queried message, with the exception of $m[i] = \blacksquare$ for any index i. Hence, $\exists i : m[i] \neq \blacksquare$, which has not been signed by theSigner. The accepting verification requires that `Sig` $= \mathsf{blkJudge}(1^\lambda, m^*, \sigma^*, pk_{sig}, pk_{san})$. Therefore, (m^*, σ^*) breaks the weak blockwise non-interactive publicly accountability. The success probability of $\mathcal{B}$ equals the one of $\mathcal{A}$.

II) ***Weak Privacy.*** *To show that Construction 14 is weakly private, there is the need to show that an adversary $\mathcal{A}$ cannot derive information about the prior content of a contained block $m[i]$, as $\blacksquare$ is considered part of the resulting message m' and all other modifications result in a forgeable RSS. Let $\mathcal{A}$ winning the weak privacy game. The construction of an adversary $\mathcal{B}$ which breaks the strong privacy experiment can be achieved in the following way:*

1. *$\mathcal{B}$ receives $pk_{san}, sk_{san}, pk_{sig}$ and forwards them to $\mathcal{A}$*

2. *$\mathcal{B}$ forwards every query to its own oracles*

3. *Eventually, $\mathcal{A}$ outputs its guess b^**

$\mathcal{B}$ outputs b^* as its own guess. The oracle requires that $\mathsf{MOD}_1(m_1) = \mathsf{MOD}(m_2)$, disregarding ■. Note, the messages are the same. Hence, the success probability of $\mathcal{B}$ is the same as $\mathcal{A}$'s.

III) ***No Strong Privacy.*** *Due to the above proof, Construction 14 is known to be weakly private. Hence, it remains to show that it is not strongly private. As a redaction leaves a* ***visible*** *special symbol, i.e., ■, an adversary can win the* ***strong*** *privacy experiment in the following way: By generating two messages m_0, m_1, where $m_1 = (m_0, 1)$. Hence, $\ell_0 < \ell_1$, while m_0 is a prefix of m_1. Afterwards, it requests that $m_1[\ell_1]$ is redacted, i.e., $\mathsf{MOD}_1 = (\ell_1)$ and $\mathsf{MOD}_0 = ()$. Hence, if the oracle chooses $b = 0$, it will output $m_2 = m_0$ and for $b = 1$, $m_2 = (m_1, ■)$. Hence, the adversary wins the game, as $(m_1, ■) \neq m_0$.* □

As existing RSSs allow for removing every block, the underlying SSS must also do this, which means it is required that ADM $= (1, \ldots, \ell)$. This rules out cases where a Signer prohibits alterations of blocks.

15.3. Minimum requirements and limitations to transform an SSS into an RSS

This section shows that standard private SSSs are not enough to build weakly private RSSs. Moreover, it provides the proof that weak blockwise non-interactive public accountability is required to build an unforgeable RSS. This — intuitive — goals get formally expressed and proven as Theorems 50 and 51.

Theorem 50 : Any non strongly private SSS results in a non-weakly private RSS.

If the SSS used in the transform is ***not*** *strongly private, the resulting RSS is* ***not*** *weakly private.*

Proof 50

Let $\mathcal{A}$ be an adversary winning the strengthened privacy game as defined in Definition 231 (see Sec. 15.1.3 on page 433). An adversary $\mathcal{B}$, which wins the weak privacy game as defined in Definition 232 (see Sec. 15.1.4 on page 434), using $\mathcal{A}$ as a black-box, is constructed as follows:

1. *$\mathcal{B}$ receives the following keys from the challenger: $\mathsf{pk}_{san}, \mathsf{sk}_{san}, \mathsf{pk}_{sig}$ and forwards them to $\mathcal{A}$*

2. *$\mathcal{B}$ simulates the signing oracle using the oracle provided*

3. *Eventually, $\mathcal{A}$ returns its guess b^**

4. *$\mathcal{B}$ outputs b^* as its own guess*

Following the definitions, the success probability of $\mathcal{B}$ equals the one of $\mathcal{A}$. This proves the theorem. □

Theorem 51 : No transform can result in a strongly private RSS.

There exists no algorithm which transforms a secure SSS into a strongly private RSS.

Proof 51

Once again, every meaningful SSS must be immutable, which implies weak immutability due to Th. 47. Hence, the following makes no statements about schemes which are not weakly immutable. One shows that any transform $\mathcal{T}$ achieving this property uses SSS′ which is not weakly immutable. Let RSS′ denote the resulting RSS. One then derives an algorithm which uses RSS′ to break the weak immutability requirement of the underlying SSS in the following way:

1. *The challenger generates the two key pairs of the SSS. It passes all keys but sk_{sig} to $\mathcal{A}$*

2. *$\mathcal{A}$ transforms the SSS into RSS' given the transform $\mathcal{T}$*
3. *$\mathcal{A}$ calls the oracle SSS.Sign with a message $m = (1, 2)$*
4. *$\mathcal{A}$ calls RSS'.Redact with $\mathsf{MOD} = (1)$*
5. *If the resulting signature σ does not verify, abort*
6. *$\mathcal{A}$ outputs $(m', \sigma_{\mathsf{SSS}})$ of the underlying SSS*

As the lengths are different, i.e., $\ell_m \neq \ell_{\mathsf{MOD}(m)}$, the message-signature pair $(\mathsf{MOD}(m), \sigma_{\mathsf{SSS}})$ breaks the weak immutability requirement of the SSS. Moreover, as hiding redacted parts of a message is essential for strong privacy, no algorithm exists, which transforms a weakly immutable SSS into a strongly private RSS, as ADM needs to be correctly recoverable. This proves the theorem. This concrete example is possible, as it uses only required behaviour. □

Theorem 52 : Weak blockwise non-interactive public accountability is required for any transform $\mathcal{T}$.

For any transformation algorithm $\mathcal{T}$, the utilised SSS must be weakly block-level non-interactive publicly accountable to result in an unforgeable RSS.

Proof 52

Let RSS' be the resulting RSS from the given SSS. Perform the following steps to show that the used SSS is not weakly block-level non-interactive publicly accountable. In particular, let $\mathcal{A}$ be an adversary winning the unforgeability game, which is used by $\mathcal{B}$ to break the weak blockwise non-interactive public accountability of the used SSS.

1. *The challenger generates the two key pairs of the SSS. It passes all keys but sk_{sig} to $\mathcal{B}$*
2. *$\mathcal{B}$ forwards all received keys to $\mathcal{A}$*
3. *$\mathcal{A}$ transforms the SSS into RSS' given the transform $\mathcal{T}$*
4. *Any calls to the signing oracle by $\mathcal{A}$ are answered genuinely by $\mathcal{B}$ using its own signing oracle*
5. *Eventually, $\mathcal{A}$ returns a tuple $(m, \sigma_{\mathsf{RSS}})$ to $\mathcal{B}$*
6. *If the resulting signature does not verify or does not win the unforgeability game, $\mathcal{A}$ and therefore also $\mathcal{B}$ abort*
7. *$\mathcal{B}$ outputs the underlying message-signature pair $(m', \sigma_{\mathsf{SSS}'})$*

Following Fig. 127, $(m', \sigma_{\mathsf{SSS}'})$ breaks the weak block-level non-interactive public accountability requirement of the SSS, as there exists a block, which has not been signed by theSigner, while theSigner is accused by blkJudge. Moreover, the success probabilities are equal. The contrary, i.e., if the SSS used is not weakly blockwise non-interactive publicly accountable, the proof is similar. To achieve the correctness requirements, the accountability definition must hold block-level. □

Theorem 53 : No unforgeable RSS can be transformed into an SSS.

There exists no transform $\mathcal{T}$, which converts an unforgeable RSS into an unforgeable SSS.

Proof 53

Let SSS' be the resulting SSS. Now perform the following steps to extract a valid forgery of the underlying RSS:

1. *The challenger generates a key pair for an RSS. It passes pk to $\mathcal{A}$.*
2. *$\mathcal{A}$ transforms RSS into SSS' given the transform $\mathcal{T}$*
3. *$\mathcal{A}$ calls the oracle RSS.Sign with a message $m = (1, 2)$ and simulates SSS'.Sign with $\mathsf{ADM} = (1)$*
4. *$\mathcal{A}$ calls SSS'.Sanit with $\mathsf{MOD} = (1, a)$, $a \in_R \{0, 1\}^\lambda$.*

5. *If the resulting signature does not verify, abort*
6. *Output the resulting signature σ_{RSS} of the underlying* RSS

As $(a, 2) \notin \mathsf{span}_{\vdash}(m)$, $((a, 2), \sigma_{\mathsf{RSS}})$ is a valid forgery of the underlying RSS*. Note, this concrete counterexample is possible, as only required behaviour is used.* □

PART IV

Related and previous work, evaluation, applications and conclusion

The eSignature Directive (Directive 1999/93/EC) has been in place for over 12 years.
The Directive ... does not cover many new technologies.

— EUROPEAN COMMISSION
Electronic identification, signatures
and trust services: Questions
and Answers, 2012 [169]

Contents

Chapter 16 Previous and related work on probative value or data protection requirements of 'non-standard' digital signatures

Chapter 17 Evaluation of the data protection capabilities and the probative value achieved by the new RSS and SSS

Chapter 18 Applications of private sanitizable and redactable signatures enabled by their increased probative value

Chapter 19 Conclusion, contributions and future work

16 — Previous and related work on probative value or data protection requirements of ‘non-standard’ digital signatures

Overview of Chapter 16

This chapter discusses related work regarding the research question, i.e. regarding the probative value of non-standard[865] digital signature schemes, especially RSS and SSS. Sec. 16.1 starts this chapter with a discussion of works that analysed the legal implications of other not standardised signature schemes. An initial overview of literature related to electronic signatures in general can be found in the works by *Laborde* [303], *Dumortier* [157], *Jungermann* [277], or in a survey from the EU [172].[866]

Then, works geared towards MSS are analysed in Sec. 16.2. This includes a discussion of previous or related work targeting the evaluation of the legal goal of data protection.

Prior to this thesis, no work on the legal implications of RSS and SSS existed as discussed in Sec. 16.3.

The chapter finally concludes in Sec. 16.4 with the observation that some of the published results gained attention and have now been accepted in the academic community, thus impacted the state of the art.

16.1. Related work in the area of legal analysis of other signature schemes

This section discusses the works from *Gennen* [203] (see Sec. 16.1.1), from *Orthacker et al.* [365] and *Heiko Rossnagel* (see Sec. 16.1.2), and from *Sorge* [451] (see Sec. 16.1.3). The works discuss other signature schemes' legal issues, however they do not discuss MSS. The following subsections are organised thematically.

16.1.1. Related work on the container signature, e.g., by *Gennen* from 2009

The work by *Gennen* [203] from 2009 discusses a technical mechanism for the communication with the courts, called EGVP[867], also known in German literature as ‘Container Signatur’ [81, 203][868]. EGVP was deployed in Germany for lawyers to securely and legally binding communicate with the courts. While the system is no longer in use today[869], when it was still in use in 2013, a court actually had to rule if the mechanism would indeed generate legally recognised electronic signatures on the documents. A brief discussion of ruling on the container signature given in Sec. 4.5.2. Technically, it is a defined

[865] Algorithm not listed in the ‘catalogues’, e.g., [75, 188], or in international standards, like RFC, ETSI, W3C or ISO, are considered non standard.

[866] This list of resources is ordered by year of publication.

[867] The German name is 'Elektronisches Gerichts- und Verwaltungspostfach (EGVP)'.

[868] This translates to 'container signature'. The BGH ruling termed it“qualifizierte Container-Signatur” [81].

[869] EGVP was in use in Germany till 2016 and has subsequently been replaced following the 'Gesetz zur Förderung des elektronischen Rechtsverkehrs' from 2013 [95].

format[870]. which describes how the container holds and structures a number of individual documents that are itself composed of several documents. The technical idea is that a valid signature on the container technically implies that all documents which are structured inside that container have not been subject to any subsequent modification.

In his work *Gennen* [203] discusses different technical aspects of the application of qualified electronic signatures on a container that is composed of multiple documents. He notes for example that the used technical process on signature generation only shows the filename of the documents in the container. *Gennen* marks this as problematic as he argues that it contradicts the requirement from § 17 Abs. 2 SigG that the document to be signed is to be shown before signing to the signatory by the signature creation component(s)[871] [203]. However, the court ruled in favour of the extended signature mechanism that was used to sign a couple of documents[872]. The BGH's ruling argued that, as the container was depending on every single documents' content, the one signature still achieved the spirit and purpose of an electronic signature to protect integrity and authenticity of the contained document [81][873].

Further legal analysis of this comment followed, e.g., by *Marly and Habermann* [329]. The authors marked in [329] that if the ruling had been different, then the existing technical solution rolled out for the electronic communication with the courts would have to be reconsidered. However, they also note that the ruling has clarified an unclear legal text, i.e., § 130 of the German ZPO a.F. [329]. Finally the authors foresee that the daily usage of a qualified signed document exchange between lawyers and courts will be eased and made more practical as a consequence[874] [329].

As such, the work of *Gennen* [203] — and of *Marly and Habermann* [329] — is related as it showed that the existing technical mechanism must be considered in depth and in its unity against the different legal aspects that describe the functional requirements for legally compliant signatures with a high probative value. The same way the methodology in this thesis first establishes the legal requirements (Part I), then describes in depth the technical properties of the mechanisms that all together form the RSS or SSS (Part III), and only then evaluates their legal impact (Part IV).

16.1.2. Related work on mobile signatures by *Orthacker et al.* from 2010 and by *H. Rossnagel* from 2004

Orthacker, Centner, and Kittl [365] discuss in their work from 2010 the Austrian approach, the so-called mobile signature, towards the requirements for qualified electronic signatures given in Directive 1999/93/EC. The authors describe the Austrian solution, which is a server based mobile signature approach, which authenticates the signatory over the trusted channel established via the signatory's mobile phone. Moreover, they discuss the merits of the signatory delegating the actual generation of the signature to a server, which holds the signatory's secret signature generation key. They identify if the mechanism fulfils the requirements on secure signature-creation devices defined by the EU directive and in particular its implementation in Austrian laws. *Orthacker et al.* [365] identified among others the following regulatory requirements from Directive 1999/93/EC:

- Article 5, 1 on the functional requirements,
- Annex I, II and III on the requirements of a secure signature-creation device, and

[870] In Germany this is the OSCI format. It was set as a government internal obligatory standard, i.e. "Das Protokoll [...] wurde vom Bundesministerium des Inneren [...] als obligatorischer Standard für elektronische Transaktionen mit der Bundesverwaltung gesetzt." [289].

[871] In German *Gennen*'s critique for the mechanism called EGVP reads as follows: "Ein nach wie vor bestehendes Problem des EGVP besteht darin, dass der Anwender vor Signatur einer Nachricht im EGVP nicht mehr überprüfen kann, welche Nachricht er gerade signiert und später versendet. Dies steht im Widerspruch zu § 17 Abs. 2 SigG, wonach Signaturanwendungskomponenten vor einer Signatur eindeutig anzeigen müssen, auf welche Daten sich die Signatur bezieht. Dazu kann es nicht ausreichend sein, den verkürzten Dateinamen einer zu Signieren [sic] in Datei anzuzeigen." [203].

[872] BGH Beschluss vom 14. Mai 2013 – VI ZB 7/13 [81].

[873] "[...] mit ihr werden Sinn und Zweck der qualifizierten Signatur — die Sicherstellung von Authentizität und Integrität des Dokuments — erreicht" [81, para. 10].

[874] "Die praktischen Folgen dieser Entscheidung dürften in einer erheblichen Arbeitserleichterung der bisher vorsichtig agierenden Juristen liegen. Die einzelnen Dateien, insbesondere bestimmende und vorbereitende Schriftsätze, müssen nun nicht mehr separat durch Verwendung einer Drittsoftware mit einer qualifizierten Signatur versehen werden. " [329].

- Annex III 1(c) on the need to allow the signatory, by means of the secure signature-creation devices, “[...] to ensure by appropriate technical and procedural means that the signature-creation data used for signature generation can be reliably protected by the legitimate signatory against the use of others.” [365].

Note, in the Austrian mechanism discussed by *Orthacker et al.* [365], the signature is actually not generated on the signatory’s mobile phone but on a server. In his work from 2004 *Heiko Rossnagel*[875] splits them into two groups: “server based and client based signatures” [418]. *H. Rossnagel* concludes that “[...] using the SIM [smart card embedded as subscriber identity module into the mobile phone] as an SSCD [...]” to generate “SIM-based signatures” is “[...] the most secure and convenient solution [...]”. Again the author uses the regulatory requirements from Directive 1999/93/EC to evaluate the legal impact of the technical mechanisms that are described in their entirety (algorithms, entities, hardware). Among others, the following parts from Directive 1999/93/EC got mentioned:

- Art.2, 2(c) regarding the need for “[...] means that the signatory can maintain under his sole control [...]” [418], or
- Annex I and Annex II regarding requirements that the certification authority must meet.

Orthacker et al. [365] further note in [365] that at that time (2010) the only published standard defining requirements for secure signature-creation devices in accordance with the directive where those related to [72, 126]. The authors state that “[...] smartcards conform to the requirements of Annex III of the directive [...]” [365]. They further remark that details of the additionally necessary surrounding mechanisms, like human interface and card readers with numeric keyboards for PIN entry can as well be found in [365]. Still, *Orthacker, Centner, and Kittl* [365] argue that “[t]he proposed server-signature scheme fulfills the requirements laid down by [Directive 1999/93/EC] on qualified signatures and provides mobile signature experience by using mobile one-time codes for authorizing the signature creation. The signature service relies on a combination of technical and organizational measures to accomplish the security requirements on technical components and processes, and the signature-creation data in particular. While storage and application of the signature-creation data is secured by technical means, the signature service partly relies on organizational measures to authorize access to components. These measures assure that the signature-creation data can be reliably protected by the legitimate signatory against the use of others.”[876] [365].

Contrary, *H. Rossnagel* [418] sees that with a signature generated by “[...] the provider it can not be verified that the customer really authorized the signature. Neither the integrity nor the fact that the user authorized it can be proven.” [418]. Thereby *H. Rossnagel* [418] argues that it is not possible to do this technically in a way compliant to the strict demands of Directive 1999/93/EC unless “[...] possible technical solutions [~~to~~] accomplish the integrity and accountability of his [the signatory’s] authorization but they would require a security environment on mobile devices that would enable the device to create qualified signatures itself [...]” [418]. In this regard *Orthacker et al.* [365] argue after citing *H. Rossnagel* [418]’s critique that “the requirement for sole control does not imply the use of special hardware as signature-creation device” [365] and that “a suitable security concept and corresponding system configuration may allow the user of a server based signature service to maintain control over his key.” [365].

In fact, it is the courts and the policymakers which decide how a compliant qualified electronic signature can be generated by the signatory. As discussed in Sec. 4.4.8 of this thesis, the latest European signature law Regulation 910/2014 now allows for those remote signatures. Note, while there has been criticisms of remote signatures and doubts of legal compliance, voiced by *A. Roßnagel* [410, 412, 414][877] or *Kment* [286], it seems that the Austrian solution discussed by *Orthacker et al.* [365] conforms to Regulation 910/2014[878], because it resembles the underlying mechanism of remote signatures. Still, time is needed to test the legal compliance. It is beyond the scope of this thesis to analyse if such a server based

[875] Not to confuse with *Alexander Roßnagel*.

[876] The American English spelling from original work is retained for terms and quotes.

[877] In 2016 *A. Roßnagel* [414] states again that in comparison to the former Directive 1999/93/EC the remote signature in eIDAS reduces the security level of a qualified electronic signature; in German “Durch diese Herabsetzung des Sicherheitsniveaus soll es möglich werden, die Signaturerstellungseinheit „der Obhut eines Dritten anzuvertrauen“, der damit Fremdsignaturen erzeugen kann, die Erwägungsgrund 52 eIDAS-VO „Fernsignaturen“ nennt.”.

[878] Taken from a server-based signature product from an Austrian company; on `https://www.xidentity.eu/xidentity/en/factx/` it was announcing that “[u]sing your digital identity by XiDentity, you can put a qualified sign anywhere, anytime” and “[t]he qualified signature conforms to the so-called “eIDAS regulation” of the European Union No. 910/2014” [last accessed: May 2017].

generation is allowed or not, but the works are related as they show again that it needs a full description and deep understanding to evaluate if a mechanism is legally compliant or not. However, the works further show that being a standardised mechanism is helpful to become favoured in a legal evaluation. Moreover, those works again discuss that legal compliance does not only consider the cryptographic scheme, but also the system in which it is used and the way the organisational controls are placed.

16.1.3. Related work on identity based signature schemes by *Sorge* from 2014

In the work by *Sorge* from 2014 [451] a special class of signatures is discussed. *Sorge* explicitly discusses “the legal classification of identity-based signatures.” [451], which is a special kind of digital signature scheme. The speciality of those signatures and thus the legally important difference are the missing public-key. Thus, the subject of evaluation of [451] is different to that of this thesis. Nevertheless, *Sorge*, in line with this thesis, states that “schemes which require private inputs as part of the signature verification process [...] cannot be used to generate qualified electronic signatures” [451]. In this respect, this thesis introduced a clear distinction between the verifiability of integrity and the authenticity and accountability of the origin. Namely, this thesis introduced non-interactive public accountability (PUB), as this form does not require any secret keys as input. With non-interactive public accountability, a scheme allows achieving technical non-repudiation only from public inputs.

Among the cited works of *Sorge* is the joint publication published in the journal ‘Datenschutz und Datenrecht (DuD)’ in 2012 [233] (see Appendix A publication n^{o} 1). The publication contains initial results[879] on the legal probative value within the legal regime of the Directive 1999/93/EC and the German SigG, which have been discussed in Sec. 4.4.2 and Sec. 4.5.

16.1.4. De-Mail law in Germany legally assigns a certain technical mechanism a high probative value

In Germany several laws were changed or passed in order to achieve a legal effect of increased probative value for a technical mechanism called ‘De-Mail’. This thesis will not go into the technical details of the whole system and the involved mechanisms itself, but rather use this as an example for a legal influence that has happened during the course of this thesis. By means of these legal changes [97, 99], the mechanism gets legally endorsed to assign a high probative value to documents that have been electronically signed within the De-Mail system. The influence of this on the general handling of authorized subsequent modifications by the German laws can be neglected (see Sec. 4.2.2.3 on page 79 for a short analysis).

The increased probative value of an electronic mail signed within the De-Mail system is codified especially through paragraph 2 of § 371a ZPO n.F.. It states that under certain conditions the electronic document sent in the De-Mail system is given a heightened legal probative value regarding its origin, i.e., the natural person that sent it. Due to being listed in the law it is automatically legally recognised and *Roßnagel* [411] states that it heightens the legal certainty[880]. The steps done in the system are as follows:

1. The natural person that is associated with the De-Mail account must be strongly authenticated, e.g. “[...] Username/Password, the German electronic ID, mobile telephone based TAN, or other secure methods”[881] [153].

[879] This thesis' analysis goes far beyond the publication [233] as the thesis also analyses the Regulation 910/2014 regime as well as the legal requirements with respect to the confidentiality offered by the cryptographic privacy property.

[880] Translated from German "De-Mail erhöht somit nicht nur die Rechtssicherheit[...]!" [411].

[881] Translated from the German original "Der Zugang zum De-Mail-Konto erfolgt über Nutzername/Passwort, über den neuen Personalausweis, mobiltelefonbasierte Verfahren (mobile TAN) oder andere sichere Verfahren." [153].

2. The De-Mail provider generating the signature on behalf of the authenticated signatory must be accredited and thus has been certified to follow certain rules when registering and authenticating users and handling user's signature generation keys[882] [97].
3. The signature generated on behalf of the natural person must be a qualified electronic signature[883] [97].

Note, a high probative value for the contents of a document is still only given by signing the contents using a qualified electronic signature [411]. It does not rule out using a MSS with equivalent strength instead of the CDSS currently used in De-Mail.

This law shows that a technical mechanism, even though its technical implementation offers a debatable[884] strength, can become legally recognised. As the De-Mail service provider, after authenticating the signatory once, will sign the electronic mail(s) on behalf of the signatory makes this service offer what the Regulation 910/2014 termed remote signatures. The additions to the law due to De-Mail however do not affect the legal impact of MSS, as the brief analysis offered in Sec. 4.2.2.3 yields.

16.2. Previous and related work in the area of legal analysis of MSS

In general a discussion of a signature scheme's confidentiality protection is obviously not possible for conventional signature schemes and thus is only found in the area of MSS. When the first work on the core topic of this thesis started around 2008, to the best of the author's knowledge, neither a discussion on the legal implications of a controlled reduced technical integrity protection as offered by RSS and SSS nor their increased confidentiality's suitability for data protection existed. The existing literature on RSS and SSS gave hints to legal requirements on the need to protect the privacy of personal data, and even pointed to concrete legislation. For example, the influential[885] work of *Ateniese, Chou, de Medeiros, and Tsudik* [12] explicitly mentions the US HIPAA as follows:

> "The additional functionalities and flexibility of sanitizable signatures may also help protect the privacy of medical records. Under the Health Insurance Portability and Accountability Act of 1996 (HIPAA), covered entities are required to comply with the Standards for Privacy of Individually Identifiable Health Information (the Privacy Rule) [...] Sanitizable signatures can be used to ensure the integrity, authenticity, and anonymity of PHI [personal health information] in both cases. In general, sanitizable signatures can accommodate different level of data de-identification, supporting the 'minimum necessary' disclosure standard of HIPAA Privacy Rule. This provides flexibility not available in redactable signatures." [12]

However, as seen above, and also in other technical works, legal texts were mentioned only to motivate the capability of RSS and SSS to offer privacy through redactions or sanitizations. Those existing works did not discuss the legal implications of the adjusted integrity and authenticity protections offered in any legal detail. At the time the research for this thesis started, there was no previous work and no related work in the area of a legal analysis of MSS. After the first publications with early results from

[882] Translated following the German "Der akkreditierte Diensteanbieter muss dem Nutzer ermöglichen, seine sichere Anmeldung im Sinne von ß 4 in der Nachricht so bestätigen zu lassen, dass die Unverfälschtheit der Bestätigung jederzeit nachprüfbar ist. Um dieses dem Empfänger der Nachricht kenntlich zu machen, bestätigt der akkreditierte Diensteanbieter des Senders die Verwendung der sicheren Anmeldung nach ß 4.[...]" § 5 Absatz 5 De-Mail-G [97].

[883] Translated following the German "Hierzu versieht er im Auftrag des Senders die Nachricht mit einer dauerhaft überprüfbaren qualifizierten elektronischen Signatur;[...] " § 5 Absatz 5 De-Mail-G [97].

[884] There was a lot of technical criticism especially due to only authenticating the signatory once, using passwords for a non-local signature and because the German government marketed a confidentiality protection while the service provider was actually always de-crypting and re-encrypting the emails. See one of the several statements issued by the German Chaos Computer Club, e.g. [149], or see `https://www.ccc.de/en/updates/2013/de-mail-unqualifizierte-makulatur` [last accessed: Sep. 2017]; all in German.

[885] *Ateniese et al.* [12] first used the term sanitizable signatures to describe the ability for authorized modifications, not just for redactions. This terminology was accepted and carried forward by other works, e.g. [64, 197, 285, 463]. Note, [264, 264, 347] contain the word 'sanitizing' in their titles, but they describe schemes that only allow the functionally of redaction. Their functionality rather follows the initial works [273, 455] and thus [264, 264, 347] get classified as RSS in this thesis (see Sec. 11.1). That's why this thesis sees the work by *Ateniese et al.* [12] as influential, both in respect to terminology and to a description of sanitization functionality.

this thesis in 2012, especially the joint journal article [233] in Datenschutz und Datenrecht (DuD) (see Appendix A publication n^o 1), other scholars picked up the results. The final Sec. 16.3 offers a summary. The remainder of Sec. 16.2 discusses in more detail the works [224, 459, 460] that are related, noting that they all appeared in the years following 2013[886].

16.2.1. Related work by *Stranacher, Krnjic, Zwattendorfer, and Zefferer* [460] from 2013

Stranacher, Krnjic, Zwattendorfer, and Zefferer [460] cite and agree with two already published joint works based on results of this thesis: [391] (see Appendix A publication n^o 6) and [233] (see Appendix A publication n^o 1). Their work from 2013 has already been influenced by this thesis's results, and hence again this indicates how relevant the contributions of this thesis are to the legal discussion.

In detail, *Stranacher et al.* identify in [460] that an RSS needs to fulfil the legal requirements of "uniquely linked to the signatory" and "capable of identifying the signatory" from Directive 1999/93/EC [175]. Namely, they state that " [...] a redactable signature scheme must satisfy the requirements of an advanced electronic signature as defined by the European Union (1999). This is a pre-requisite for accountability and to identify the original signer." [460, p. 510].

This is in line with this thesis and the joint works cited. The authors further state that an RSS may only "optionally" [887] [460, p. 510] fulfil the requirements of qualified electronic signatures [175].

In line with the thesis analysis Requirement 4[888] *Stranacher et al.* further states that an RSS must enable accountability[889]. This thesis noted that a missing public accountability, i.e., no non-interactive public accountability, is not just a minor issue, but would limit the legal certainty (see Subsections 6.4.2 and 6.4.5). However, this thesis wanted to establish a deeper understanding of what the property of transparency does. Contrary, *Stranacher et al.* noted "Pöhls et al. (2011) contains only minor updates on the property transparency (which is not of special interest for our use cases)." [460] citing the results published jointly in [391]. The authors further state that an RSS must enable designated redactors[890]. With designated redactors the RSS becomes Sanitizer-accountable. In the same line this thesis showed how to add accountability for redactions (see Sec. 14.1 for the framework of ARSS). The construction $pubacc\mathcal{RSS}$ achieves public non-transparent accountability for RSS (see Sec. 14.14). The cryptographic definition and further details are in Sec. 14.1. Note that, while talking about "redactors" [460] *Stranacher et al.* [460] actually later in their work analyse only MSS which offer sanitization functions, e.g. sanitizable schemes like that of *Ateniese et al.* [12]; the name is thus misleading. Accountability was not among the properties of common RSS schemes. In particular accountability was not mentioned as a property in the cryptographic literature of RSS at that time; It was generally accepted that in an RSS everybody can take the role of the redactor. In this light, *Stranacher et al.* equally acknowledge that accountability cannot be met by an RSS, but only by SSS as they correctly state that:

> "[...] these schemes [RSS proposed in [455], [442], [117], [66]] do not support the specification of designated redactors [...]" [460, p. 511]

This thesis obtained accountability for RSS and the resulting ARSS (see Sec. 14.1) has also been published as joint work (see Appendix A publication n^o 21).

[886] This thesis proudly notes that those works are dated post the publication date of joined works; thus the results of this thesis were indeed unique at the time of publication, i.e., in 2012. Of course the works do cite the joint works.

[887] "[...] a redactable signature may, optionally, meet also the requirements for qualified electronic signatures as defined by the European Union (1999)." [460, p. 510].

[888] Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4).

[889] "**General legal requirements** [...]
Accountability: In case of a dispute the signatory must be able to prove that certain modifications have been done by a certain redactor. Accountability can be achieved by technical means (see also technical requirements below)" [460, p. 510]

[890] "**Technical requirements** [...]
Designated Redactors: Designated redactors must be able to be specified by the redactable signature scheme. That means that the signatory must be able to determine who is allowed to modify the signed data. Persons except the signatory and the designated redactors must not be able to redact data without breaking the originally signature applied. Any change of the data by unauthorized persons must be recognizable." [460, p. 510]

This thesis separated secure RSS from secure SSS with great care (see Chapter 15). Following this separation, the following statement from *Stranacher et al.*'s work must be flagged: "To remove or replaced [sic] designated parts of the signed messages with an arbitrary string, Ateniese et al. (2005) proposed Sanitizable signatures." [460, p. 511]. This thesis assumes that *Stranacher et al.* has the same understanding of the sanitization functionality offered by MSS as this thesis, because they also cite *Ateniese et al.* [12]. Then the above statement is problematic as it weakens the separation between the redaction and the sanitization functionality because it mixes the remove-only or replace-with-■ functionality offered by RSS with the sanitization functionality of SSS, which allows for arbitrary replacements. However, at the time of *Stranacher et al.* [460]'s publication (2013) the results of the separation [388] were not published (2015). However, this thesis finds at least the following statement by *Stranacher et al.* misleading: "Sanitizable signatures can be seen as a small subset of redactable signatures, as they are basically redactable signatures where the replacement part is permanently exchanged." [460, p. 511]. This thesis considers this a miss-interpretation of the sanitization functionality as presented by *Ateniese et al.* [12][891].

In line with this thesis requirements and understandings is that *Stranacher et al.* also note that a practical RSS must be private. *Stranacher et al.* give a definition of privacy which is in line with this thesis's definition of standard privacy given in Definition 156 for SSS and in Definition 174 for RSS.

Further, the authors identify that an RSS must be able to allow the signer mark those designated parts that can be redacted. For this thesis this explicit Signer control is part of requested flexibility, legally it would be enough if the Signer consent to the offered protection, or weakened protection of integrity can be assumed.

Finally, *Stranacher, Krnjic, Zwattendorfer, and Zefferer* require that an RSS must be "[a]pplicable" [460] and "[c]ompatible" [460], by which *Stranacher et al.* mean that the RSS can handle tree-based documents, like XML, and supports XMLDSIG as an existing signature standard. Again, *Stranacher et al.* cite an already published result of this thesis [391]: This thesis discusses in Sec. 18.1 how an XML embedding of MSS can be accomplished. Those XML structures are based on the results that got published as joint work with *K. Samelin* and *J. Posegga* [391] (see Appendix A publication n^{o} 6).

16.2.2. Related work by *Stranacher, Krnjic, and Zefferer* [459] from 2013

Stranacher, Krnjic, and Zefferer [459] mention shortly the EU regulation on signatures in their work on the application of RSS in the public sector from 2013. They do this to motivate that "[...] electronic signatures are widely used in transactional e-government processes [...]" [459] and that there is "a common legal basis" [459]. However, they do not provide an analysis towards legal requirements for a probative value nor make statements regarding their fulfilment by RSS in [459].

16.2.3. Related work by *Zwattendorfer and Slamanig* [513] from 2013

Zwattendorfer and Slamanig [513], discuss the application of redactable signatures in relation to their integration into the Austrian eID system. They analyse, among other cryptographic solutions, if an RSS (in combination with proxy re-encryption) can be used to move the authentication service, called MOA-ID, into a public cloud and still "[...] preserve citizens' privacy when assuming that MOA-ID acts honest but curious." [513]. They do find that the approach using RSS "[...] might be the best as it could be quickly realized and requires less effort for the client-side middleware and the service provider. However, linkability and higher efforts for extensions are the drawbacks of this approach."[892] [513, 514]. *Zwattendorfer and Slamanig* [514] do state that "[...] MOA-ID in the cloud must still work equivalent to the current Austrian eID system and must still be compliant to the Austrian national law." [514]. However, the authors do not analyse the legal aspects of the solutions resulting from the usage of a redactable signature nor provide legal requirements for it in [513]. A legal analysis is also not presented in later work from *Zwattendorfer* et al. published later in 2015 [514] and 2016 [515] — all works discuss the technical use of redactable signatures in the Austrian eID.

[891] *Stranacher et al.* reference [12] to describe sanitizable signatures.

[892] The American English spelling from original work is retained for terms and quotes.

16.3. No related work existed on the legal implications of MSS

Prior to the results published in the course of this thesis there was no legal analysis of the reduced integrity protection offered by redactable or sanitizable signature schemes. Foremost, there was also no integrity 'metric' to compare the schemes among each other and contrast them with classical, legally accepted ones. Previous work was carried out for other specialised non-standard signature schemes but neither for RSS nor SSS as discussed in Sec. 16.1. *Slamanig and Rass* [443] noted in 2011 in a German journal article that such an analysis would be very valuable for a practical usage[893].

This gap is filled by a number of publications that timely disseminated the results obtained on the legal implications during the course of this thesis' analysis:

- The joint work with *F. Höhne* published at IFIP STM 2011 [381] offers a first — partial — analysis of the legal integrity definitions in the light of authorized subsequent modification. The classification into *CA and *CD are presented (see Appendix A publication n^{o} 7).
- The new cryptographic property of non-interactive public accountability (PUB) is formally described and published as joint work with *C. Brzuska* and *K. Samelin* at EuroPKI in 2012 [68] (see Appendix A publication n^{o} 11).
- To directly answer *Slamanig and Rass* [443], the analysis — jointly done with *F. Höhne* — of the legal implications of redactable and sanitizable signatures on the basis of the German SigG is published in the same journal as [443] — Datenschutz und Datenrecht (DuD) — in 2012 [233] (see Appendix A publication n^{o} 1).
- The usefulness and legal impact of the application of RSS in food supply chains is published as joint work with *F. Höhne* at the 5th Interdisciplinary Conference on Current Issues in IT Security of the Max Planck Institute for Foreign and International Criminal Law in 2012 [382] (see Appendix A publication n^{o} 9).
- The legal implications of MSS as signatures in the light of Regulation 910/2014 were analysed in 2015 [199] as joint work with *F. van Geelkerken* and *S. Fischer-Hübner* (see Appendix A publication n^{o} 22).

The acceptance for publication of the results by the legal scholars interested in technical aspects and at venues interested in interdisciplinary aspects of IT law confirms that such a discussion was indeed missing and of interest to the community. The results on the legal implications of using in particular MSS from this thesis fill the gaps in legal, cryptographic as well as in interdisciplinary research. Obviously, the state of the art is advanced when results published during the course of this thesis get used and cited, e.g., [224, 460]. More examples are presented in the next section.

16.4. Impact on the state of the art

The adoption of published results by other scholars shows the influence on and the advancement of the state of the art. In the following this thesis lists some examples showing the adoption of the cryptographic properties devised in the course of the thesis and published after academic peer review:

Stranacher, Krnjic, Zwattendorfer, and Zefferer [460] cite and agree with the works published in ACNS in 2010 [391] (see Appendix A publication n^{o} 6) and in the journal Datenschutz und Datensicherheit in 2012 [233] (see Appendix A publication n^{o} 1). This was already discussed in detail in Sec. 16.2.1.

Slamanig, Stranacher, and Zwattendorfer [445] state: "In case that it should be publicly verifiable that the modified message $\hat{m}$ has been produced by some particular redactor (non-interactive public accountability [[68]]), the redactor is required to create a conventional digital signature on it."[894] [445] Their work cites the work on non-interactive public accountability [68] (see Appendix A publication n^{o} 11).

[893] In the German original: "Ein Aspekt, der hier keine Berücksichtigung fand, jedoch für eine praktische Verwendung sehr relevant scheint, ist eine juristische Betrachtung von redigierbaren Signaturen." [443].

[894] Original's endnote has been adjusted to correctly point to the same entry found in the bibliography of this thesis.

Fleischhacker, Krupp, Malavolta, Schneider, Schröder, and Simkin's work published at PKC 2016 [197] (also published as a pre-print in 2015 [196]) cites the following published papers containing results on non-interactive public accountability from this thesis: [68, 69, 387][895]. The notion of non-interactive public accountability has been accepted in MSS literature showing that a differentiation (between PUB and INT) was missing, or as *Fleischhacker et al.* [197] explain with reference to [68] (n^o 11): "[...] non-interactive accountability and transparency are mutually exclusive. That is, no scheme can fulfill both properties at the same time. In this work we focus on schemes that have (interactive) accountability and transparency."[896] [196].

Lai, Zhang, Chow, and Schröder [305] presented their work at ESORICS 2016 citing the joint works published at EuroPKI in 2012 [68] (n^o 11) and 2013 [69] (n^o 17) adopting again the thesis' result of differentiating correctly between the public form of accountability (PUB) and transparency. They restate correctly "[...] which allows a third party to determine if a message originates from the signer or the sanitizer, without any help from the signer." [305] citing the joint works. Further, the authors also acknowledge the thesis' classification and relations to the cryptographic property of transparency, stating that "[...] non-interactive accountability and transparency cannot be achieved simultaneously [...]" [305] while one can "[...] have (interactive) accountability and transparency." [305]. Further, the strict notion of unlinkability that "[...] also conceal[s] the sets of allowed modifications [...]" [305], as introduced in the joint work published at EuroPKI 2013 [69], got acknowledged and discussed as "[...] difficult to construct such a scheme efficiently." [305]. The authors of [305] then point to the work done by *Derler and Slamanig* [139].

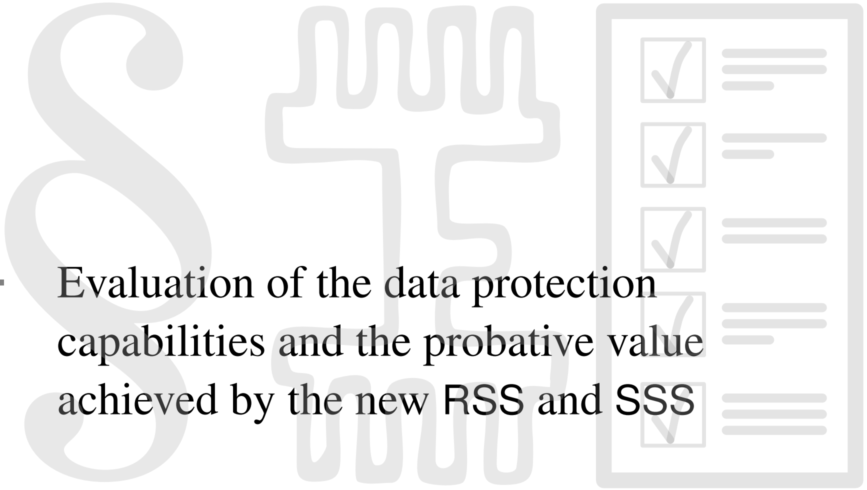
Derler and Slamanig [139] in their work from ProvSec 2015 note that the strengthened notion of strong unlinkability as presented in Sec. 13.3 — and published jointly at EuroPKI 2013 — of this thesis is helpful as "[...] a stronger notion of privacy, i.e., (strong) unlinkability (defined by *Brzuska et al.* at EuroPKI'13) [...] fixes this problem [...]"[897] [139]. Their work elaborates on how to construct such schemes more efficiently while weakening the security only slightly.

Also very recently in 2017, some of the several schemes and properties being proposed and published are acknowledged and cited as related work, for example the $struct\mathcal{RSS}$ [426] (see Sec. 14.9) and $priv\text{-}\mathcal{RSS}$ [425] (see Sec. 14.5) is acknowledged by *Erwig, Fischlin, Hald, Hehm, Kiel, Kübler, Kümmerlin, Laenge, and Rohrbach* [165] in their work on redactable graphs published at ACISP 2017.

[895] See Appendix A publication n^o 20 [387], n^o 11 [68], and n^o 17 [69].

[896] The American English spelling from original work is retained for terms and quotes.

[897] The reference "Brzuska et al. at EuroPKI'13" [139] identifies the joint work of *Brzuska, Pöhls, and Samelin* [69] (see Appendix A publication n^o 17).



17 Evaluation of the data protection capabilities and the probative value achieved by the new RSS and SSS

Overview of Chapter 17

This thesis used, among others, the legal texts depicted in Fig. 128. The figure additionally shows their position within the Hierarchy of Norms. The analysis has provided that a signature scheme, in order to give a high probative value to the signed document, has to achieve the technical requirements listed in Chapter 7. Additionally, the malleable signature scheme would need to protect the confidentiality of the removed data from prying eyes in order to become a suitable technical instrument for legally compliant data protection. The technical requirements towards the cryptographic property of privacy are listed in Chapter 10. All so-far elicited requirements are checked for the schemes proposed in this thesis in this chapter. Especially, two proposed schemes which fulfil all recommended requirements are selected and a final legal evaluation for them is presented to answer the research question. To recall, it was as follows:

> Can a malleable signature scheme be private to be compliant with EU data protection regulation **and at the same time** fulfil the integrity protection legally required in the EU to achieve a high probative value for the data signed?

The positive evaluation results are

- Evaluation Result 1: All proposed schemes from Chapters 13 and 14 fulfil data protection requirement (Req. 9) as they achieve standard privacy,
- Evaluation Result 2: MSS are ‘elementary’ electronic signatures following Regulation 910/2014,
- Evaluation Result 3: A signature by an MSS with ACA – ≥1CD – PUB, that yields that the signed document is unmodified, is a qualified electronic signature following Regulation 910/2014,
- Evaluation Result 4: A signature with an MSS with ACA – ≥1CD –PUB integrity but no Sanitizer-accountability generates a legal blanket statement,
- Evaluation Result 5: Rules to award evidence with a legal presumption of authenticity carry over to MSS,
- Evaluation Result 6: MSS can offer detectability of double sanitized documents created by sanitization or redaction from the same origin, but no ordering can be derived, and
- Evaluation Result 7: MSS can be facilitated to prove that a different (or original) version of a sanitized document exists, assuming one is in possession of that version’s message-signature pair.

Essential results from the above mentioned legal evaluations have been disseminated to the academic audience in three joint publications:

- in 2011 by *Pöhls and Höhne* [381] with a focus on the definitions of integrity with respect to authorized subsequent modifications (see Appendix A publication nº 7),
- in 2012 by *Höhne, Pöhls, and Samelin* [233] with a focus on Directive 1999/93/EC (see Appendix A publication nº 1), and
- in 2015 by *F.W.J. van Geelkerken, Pöhls, and Fischer-Hübner* [199] with a focus on Regulation 910/2014 (see Appendix A publication nº 22).

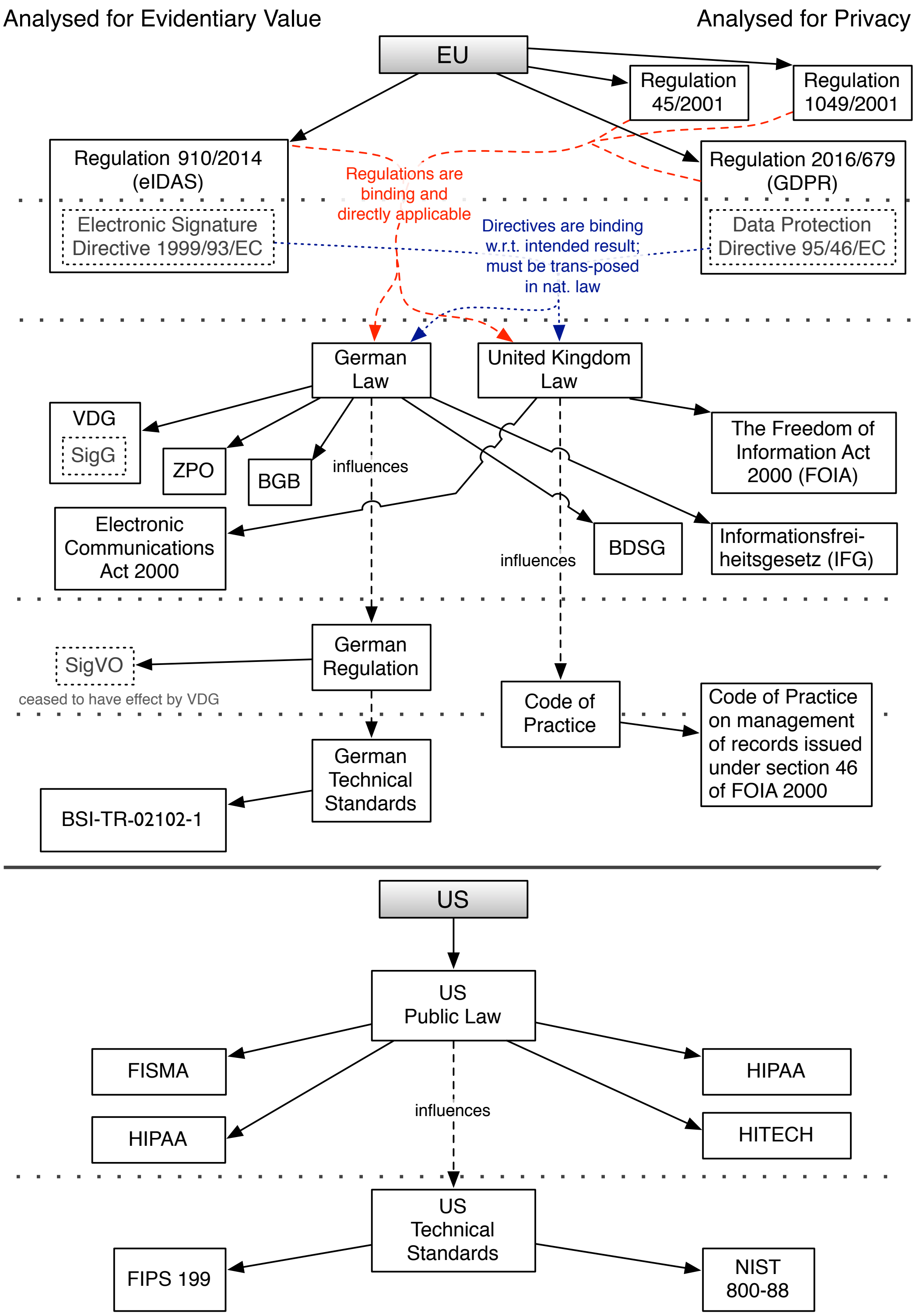

Analysed for Evidentiary Value
Analysed for Privacy
EU
Regulation 45/2001
Regulation 1049/2001
Regulation 910/2014 (eIDAS)
Electronic Signature Directive 1999/93/EC
Regulations are binding and directly applicable
Directives are binding w.r.t. intended result; must be trans-posed in nat. law
Regulation 2016/679 (GDPR)
Data Protection Directive 95/46/EC
German Law
United Kingdom Law
VDG
SigG
ZPO
BGB
influences
The Freedom of Information Act 2000 (FOIA)
Electronic Communications Act 2000
BDSG
Informationsfrei-heitsgesetz (IFG)
influences
German Regulation
SigVO
ceased to have effect by VDG
Code of Practice
Code of Practice on management of records issued under section 46 of FOIA 2000
German Technical Standards
BSI-TR-02102-1
US
US Public Law
FISMA
HIPAA
HIPAA
HITECH
influences
US Technical Standards
FIPS 199
NIST 800-88

Figure 128. Overview and hierarchy of legal norms; Left: those used to identify the probative value of malleable signatures; Right: those used to assess the privacy requirements

17.1. Evaluation of data protection capabilities of proposed schemes (Req. 9)

The confidentiality protection of personal data is legally important (see Sec. 8.4.3 stating this as Analysis Result 16[898]). Subsequent redactions or overriding modifications are a technical and legally accepted method to remove personal data (see Sec. 8.4.6 stating this as Analysis Result 19[899]). Private RSS and SSS offer data minimisation capabilities by carrying out data minimisation through redactions or sanitizations on signed personal data as defined in Definition 124 on page 202:

Definition 124 : Data Minimisation Principle following EU GDPR

> *The principle of data minimisation dictates that the personal data collected must be "[...] adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed" [Art. 5 GDPR].*

Previously, in the sections on the confidentiality requirements, this thesis left it open to define the legally sufficient strength in terms of cryptography. This section fills this gap and finally evaluates the needed strength of the cryptographic property of privacy for RSS and SSS to become legally compliant for data protection applications. After a short discussion of the benefits (see Sec. 17.1.1) and suitability (see Sec. 17.1.2) the requirement for achieving standard security is formulated (see Sec. 17.1.3).

17.1.1. MSS uphold the integrity protection, which is beneficial

If a cryptographically private MSS was used to sign a document containing personal data — or any other data in need of later modifications — , then the loss of authenticity occurring with classical signature schemes, once a block of the signed document is removed, would not occur. This is an advantage of MSS over a plain digital signature scheme. Note that the Regulation 2016/679 (GDPR) demands to uphold the integrity of personal data as well.[900] Thus, NCA–integrity-protected personal data might be more seldom subject to deletions for data minimisation compared to ACA–integrity-protected one. Because with NCA–integrity protection, the before-mentioned legal requirement from GDPR — to keep a verifiable integrity protection — is in conflict with any subsequent modification — that would break the verifiable integrity — due to data minimisation for an increased data protection. MSS that offer ACA-integrity protection would allow to unite these perceived incompatible goals and allow the data holder to uphold both: keep a verifiable integrity and do subsequent modifications for data minimisation.

Further, ISO 27042 concurs with the need for modifications by redaction and sanitization, but raises concerns that it could be a potential problem for digital evidence. The ISO standard acknowledges that there are situations when "[...] information [...] must not be disclosed to some communities [...]" [256]. This was stated as Analysis Result 19[899]. On the one hand, the analysis points out the relation to ISO 27038[901] — especially that for a successful "redaction" [258] care must be taken to ensure that the "[...] redacted information is permanently removed from the digital document [...]" [258]. The analysis in Sec. 9.3.5 has shown that ISO-27038 does not discuss the redaction or sanitization of integrity or authenticity-protected digital documents. There is a gap in ISO standardisation[902]. On the other hand, ISO 27042 clearly highlights a potential negative impact of the subsequent modifications: They "[...] can affect one's ability to investigate by introducing obfuscation mechanisms." [258].

This thesis offers an analysis of the probative value of MSS and hopes to aid establishing MSS as one suitable tool — potentially among others — to solve such a digital evidence sanitization problem.

[898] Analysis Result 16: Legal texts acknowledge the importance of confidentiality protection of personal data and the high-level descriptions of the goal of confidentiality coincide (see Sec. 8.4.3 on page 208).

[899] Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism (see Sec. 8.4.6 on page 211).

[900] The GDPR demands "[...] ensur[ing] the ongoing confidentiality, integrity [...]" [Art. 32, Item (b) GDPR] and "[...] ensur[ing] appropriate security of the personal data, including protection against unauthorized or unlawful processing and against accidental loss, destruction or damage, using appropriate technical or organisational measures ('integrity and confidentiality')." [Art. 32 Item (b) GDPR].

[901] ISO 27038 has the title 'Specification for digital redaction' [256].

[902] The author of this thesis is currently undertaking activities to jointly initiate an ISO standard for 'Redaction of Authentic Data'; a so-called new work item proposal (NWIP) was proposed in Nov. 2017; this is an early preliminary stage (00.20) within the ISO standardisation process; see `https://www.iso.org/stage-codes.html` [last accessed: Nov. 2017] for an overview of the stages within ISO.

17.1.2. Private MSS are a technical tool to minimise data, which is beneficial

As discussed in Sec. 8.2.7 on page 202, one principle from Article 5 GDPR is to minimise the amount of personal data that is processed. Note that the term 'processing' in GDPR includes also the communication and storage of data. In light of the data minimisation principle the removal of unnecessary information from data, *before* storing or communicating it, becomes a legal necessity. The German § 3a BDSG additionally contains a clause that is balancing the respective interests of the data processor in terms of overhead.[903] According to ENISA, reducing the amount of personal data helps "[...] reducing the need for [...] securing stored data." [187].

While ISO 29100 does not contain the terms 'redaction' nor 'sanitization', the standard's section on privacy-enhancing technologies (PET) contains similarly worded operations: "[...] eliminate, reduce, mask, or de-identify PII [...]" [250]. Quite explicitly masking gets defined as "[...] the process of obscuring elements of PII." [250]. ISO 29100 describes that blocks of a message, which maps to "elements of PII" [250], can get masked or eliminated individually. Thus, this thesis follows that ISO 29100's goal of an "elimination" [250] can be achieved when the redaction removes the information with strong enough cryptographic privacy guarantees.

The data minimisation achieved by an MSS can also be economically beneficial. By redaction or sanitization the amount of potentially exposed personal data in the case of an infringement is reduced. Noteworthy, the GDPR has increased the potential fines[904] for privacy breaches compared to previous legislation[905]. Thus, the use of MSS can even become economically viable: The amount of a potential fine, following GDPR Article 82–84, is determined by several criteria, among them "[...] the degree of responsibility of the controller or processor taking into account technical and organizational measures implemented [...]"[906] [Art. 83, Par. 2, Item (d) GDPR]. Here, a reduction of the amount of authenticated data which can potentially be compromised is made possible if an MSS allows for the redaction and sanitization. The MSS can thus be seen as a technical measure suitable to reduce or avoid a potential fine.

17.1.3. Data protection requirement: Achieve at least cryptographic standard privacy following *Brzuska et al.*

As the removal of unnecessary information, if done with sufficient protection against recovery, also reduces the risk of it being breached, the removal lowers the risk and the costs associated with it. Data minimisation only works against adversaries if the protection against recovery of the previously redacted or sanitized parts is cryptographically strong. This was formulated as Requirement 9[907].

As per Requirement 9 this thesis requires MSS to be cryptographically private as formally defined following *Brzuska et al.* [64] with the adjusted definition of standard privacy in Definition 156 for SSS and in Definition 174 for RSS. Informally the cryptographic property of standard privacy has been defined as follows:

Definition 155 : Standard Privacy

> *A **private** scheme prevents Verifiers from recovering any information (esp. the original value) about block(s) of m, which are no longer in the sanitized block(s) of m', through a valid signature σ' over m'.*
>
> *Note: Information leakage through the semantic content of the modified message m' itself, which is given to the adversary, is out of scope.*

[903] "In particular, personal data are to be aliased or rendered anonymous as far as possible and the effort involved is reasonable in relation to the desired level of protection." [91].

[904] See Art. 83 GDPR and Recitals 148–152 GDPR.

[905] "Fines from the Information Commissioner's Office (ICO) against Brit[ish] companies last year would have been £69m rather than £880, 500 if the pending General Data Protection Regulation (GDPR) had been applied, according to analysis by NCC Group." [311].

[906] The American English spelling from original work is retained for terms and quotes.

[907] Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength) (see Sec. 10.1).

The formal definition of standard privacy can and have been formulated analogue to the indistinguishability notion, this fact has been recorded as Analysis Result 20[908]. This is the basis for the definition from *Brzuska et al.* [64] which gives the blackbox intuition of an adversary who can not differentiate if a block was removed before signing or afterwards by redaction or sanitization as depicted in the Fig. 21[909]. Note, the high-level presentation omits details like the specification of the admissible modifications by providing the authorized modifications policy (ADM) for each adversary provided message. Of course the cryptographic games provided in the definitions of the property (Definition 156 and Definition 174) allow this and also take care that the adversary can not cheat by enforcing error conditions, e.g. the blackbox always checks that for both messages and their corresponding policy the corresponding adversary's modification instruction is actually allowed.

The definitions given in this thesis are based on the works of *Brzuska et al.* and include the updates incorporating the scrutiny of *Gong et al.* [216] as well as clarifying additions following *Krenn et al.* [294]. For more details on the cryptographic definitions please see the discussion offered in Sec. 11.6.3 on page 263 for Definition 156 for privacy in SSS and respectively in Sec. 11.13.3 on page 288 for Definition 174 for privacy in RSS.

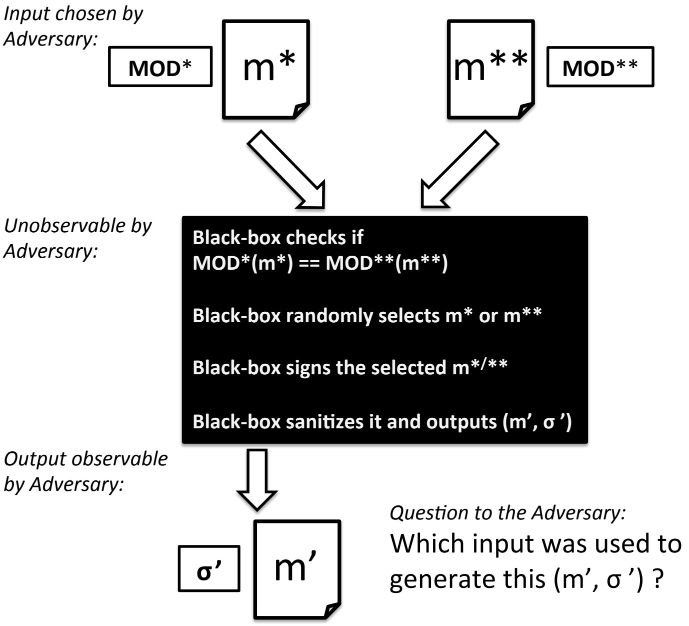

Input chosen by Adversary:
MOD*
m*
m**
MOD**
Unobservable by Adversary:
Black-box checks if MOD*(m*) == MOD**(m**)
Black-box randomly selects m* or m**
Black-box signs the selected m*/**
Black-box sanitizes it and outputs (m', σ ')
Output observable by Adversary:
σ'
m'
Question to the Adversary:
Which input was used to generate this (m', σ ') ?

Figure 21. High-level view of the privacy game (following [64]): The adversary should not be better than guessing when deciding which of the adversary provided input message produced the black-box's output.
(same Figure as on page 60)

Fulfilling the cryptographic property of standard privacy — following its definition in Definition 155 — inhibits an adversary that is capable of adaptively choosing all message-related inputs except the key pairs[910] to revert any modification done through a correctly executed sanitization by a Sanitizer. It is this achievement of confidentiality through the cryptographic property termed 'privacy' that makes malleable signatures a valuable tool for data protection. The minimal level is standard privacy as it allows the RSS/SSS the blackening/anonymization of personally identifying information to conform with data protection rules, or protect the confidentiality of trade secrets, while simultaneously being able to share and distribute the remaining information from signed documents with verifiable integrity and authenticity. Note, no leakage occurs through the signature if a scheme is standard private an adversary has no advantage over guessing when trying to identify which of two adversary supplied structurally

[908] Analysis Result 20: The definition of the cryptographic property of privacy based on indistinguishability provides a well defined strength of confidentiality and a starting ground for a definition of standard privacy. (see Sec. 9.4.1 on page 233).

[909] The same figure was already used during the introduction of the privacy property in Sec. 3.9.

[910] The adversary can adaptively choose messages m, the admissible modifications ADM, and the sanitizer via $\mathsf{pk}_{\mathrm{san}}$; it can not choose $\mathsf{pk}_{\mathrm{sig}}$ and especially not $\mathsf{sk}_{\mathrm{san}}$. Even in *strengthened* privacy as defined in Sec. 15.1.3 the adversary can not choose $\mathsf{sk}_{\mathrm{san}}$, but only knows it.

equal input messages was used to derive a valid sanitized/redacted message. Where structurally equal means that the message resulting from the sanitization of each of the two adversary supplied messages must not allow to leak the choice of message from the remaining immutable blocks, from the remaining structure or from the remaining admissible modifications (ADM'). The research conducted for this thesis, and works by others, show that several existing schemes, claimed to provide such a leakage-freeness, are not private under *Brzuska et al.*'s definition (see Shortcoming 1 in Sec. 12.1 on page 294).

To summarise, MSS achieving cryptographic standard privacy following *Brzuska et al.* do prevent leakage of sanitized information through the sanitized signature value. Only when an MSS prevents this leakage from occurring they can be used as a technical tool to carry out sanitizations and redactions achieving the legally indicated results (see Analysis Result 19[a]). Achieving standard privacy from Definition 156 or Definition 174 is one way to formally prove that the known attacks are prohibited — and thus no information about the previous contents of the sanitized data (or structure) leaks from the derived signature. Note again, leakage through other means is still out-of-scope[b].

[a] Analysis Result 19: Document sanitizations (including redactions) are a legally recognised confidentiality protection mechanism (see Sec. 8.4.6 on page 211).

[b] See also Sec. 9.4.3 stating Analysis Result 22: Leakage through the message is out of scope of MSS properties on page 233.

17.2. Evaluation Result 1: All proposed schemes from Chapters 13 and 14 fulfil data protection requirement (Req. 9) as they achieve standard privacy

This thesis has elicited the following two technical requirements for a sufficient confidentiality protection — the second being optional:

- Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength)
- Requirement 10 (Mechanism MAY prohibit the identification of additional context information further to Requirement 9 without violating Requirement 4)

All eight proposed schemes devised in the course of this thesis (described in Chapter 13 and Chapter 14) fulfil at least standard privacy as defined in Definition 174 or Definition 156. Both definitions are based upon and as strong as the one given by *Brzuska et al.* [64]. The cryptographic proofs are found in the respective sections and these results have also been published. The overview is given in Tab. 27.

Proposed Scheme (Data Structure)	**Requirement 9: Standard Privacy (Def. 174 & 156) following *Brzuska et al.***	**Stronger privacy?**	**Proof of privacy**	**Publication**
pubacc$\mathcal{SSS}$ (list)	✓	–	Sec. 13.5.3	[68]
blockacc$\mathcal{SSS}$ (list)	✓	optional transparency[911]	Sec. 13.7.4	[132]
unlinkable$\mathcal{SSS}$ (list)	✓	perfect privacy and strong unlinkability	Sec. 13.9.4	[69]
priv$\mathcal{RSS}$ (tree)	✓	–	Sec. 14.6.4	[425]
flex$\mathcal{RSS}$ (tree)	✓	transparency	Sec. 14.8.4	[134], [394]
struct$\mathcal{RSS}$ (list)	✓	transparency (Sec. 14.10.4)	Sec. 14.10.4	[426]
merge$\mathcal{RSS}$ (set)	✓	–	Sec. 14.12.4	[393], [387]
pubacc$\mathcal{RSS}$ (list)	✓	optional transparency[912]	Sec. 14.14.3	[388]

Table 27. All schemes proposed in this thesis are practically usable for protecting private data, because they are — at least — private as defined by *Brzuska et al.* [64]; this fulfils Requirement 9

[911] *blockacc*$\mathcal{SSS}^{INT}$ achieves transparency.

[912] For transparency it is required that the SSS and the RSS used as building blocks are transparent, then *pubacc*$\mathcal{RSS}$ achieves transparency (INT) instead of non-interactive public accountability (INT).

All eight proposed schemes (and all ten constructions) fulfil Requirement 9. In detail some of the new schemes can even achieve stronger privacy notions, i.e., transparency or unlinkability and thus fulfil Requirement 10. Following the analysis given in (see Shortcoming 1 in Sec. 12.1) being at least standard private allows the MSS to be used in practice as a tool for data protection or allows cryptographically sound protection of trade secrets once they have been subject to sanitization (i.e., modified to a hiding value in SSS or redacted in RSS).

The above argumentation is based on the technical qualities of the algorithm, i.e., the cryptographic strength. If disputed in court it would require an expert opinion, part of which is the technical, i.e., cryptographic, strength discussed in this thesis. Note, as mentioned in Sec. 17.8.1 this thesis is not a full expert opinion, this would require to take into account the exact software implementation, hardware used and operational procedures. This thesis however aims to be a basis for the technical expert opinion regarding the proposed algorithms cryptographic strength to protect the confidentiality.

Note, it is out of the scope of the technical schemes and this thesis to identify what information needs to be sanitized to protect personal data or trade secrets. This is application knowledge. A suitable technical scheme must be chosen and then configured according to it. In particular the application needs to configure the decomposition into blocks and the policy. Equally the choice of which Sanitizer will be authorized needs to be made by the Signer based on the application to support.

Note further that if information is encoded in the data's structure, then a scheme capable of removing or sanitizing the structure, like $flex\mathcal{RSS}$ (see Sec. 14.7), should be considered during signature generation to enable flexible redactions, e.g. only structure and not content. All proposed schemes do not make any assumptions on the semantic contents of the message and it is assumed that messages of any type can be encoded securely into the cryptographic message space of $\mathcal{M} = \{0,1\}^*$. See the XML message conversion devised and used within ReSCUeIT (see Sec. 18.1.3) as an example of such an encoding from XML into a set of strings.

17.3. Evaluation of probative value of all schemes (Req. 1–7)

The evaluation for the increase of probative value checks the fulfilment of Requirements 1–7 as they have been distilled from the legal texts from Germany and the European Union that would provide an electronically signed document with an increased probative value. The legal texts analysed for this thesis show that within the EU a signature generated by a mechanism in compliance with the requirements for *qualified electronic signatures* according to Regulation 910/2014 will allow to gain prima facie evidence or even evidence with a legal presumption of authenticity following the assignment rules found in the German ZPO. That the legal status and thus the probative value is not clear for any technical mechanism below a *qualified* one was confirmed in the work by *Grigorjew* [219]. To avoid not being classified as a qualified signature the requirements (Req. 1–7) have been crafted such that by fulfilment the respective MSS demonstrates that the scheme's construction is technically equivalent, except for providing ACA integrity instead of NCA. Namely, with non-interactive public accountability (PUB) the MSS can be aligned to a classical digital signature scheme (CDSS) such that the only difference is ACA versus NCA. This section evaluates the proposed schemes towards their fulfilment of each single technical requirement for an increased probative value. Tab. 28 provides an overview of the seven mandatory technical requirements — Requirement 8[913] is optional — for an increased probative value. In Sec. 17.5 the achieved alignment between MSS and CDSS is used to argue for the increased probative value offered by MSS with the correct set of cryptographic properties.

[913] Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional)).

Requirement	offered by an MSS?	Section	Notes
Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer)	✓	17.3.1	All proposed schemes fulfil this due to being unforgeable and immutable.
Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier)	✓	17.3.2	All proposed schemes fulfil the requirement's mandatory parts because the signature fixes scheme and policy.
Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification)	✓	17.3.3	All proposed schemes fulfil the requirement's mandatory parts because any subsequent modification is detected ($\geq$1CD), either INT or PUB.
Requirement 4 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer; Mechanism MAY achieve non-interactive public accountability of the Sanitizer)	✓	17.3.4	$\mathit{pubacc}\mathcal{SSS}$, $\mathit{pubacc}\mathcal{SSS}^{\mathit{Block}}$, $\mathit{blockacc}\mathcal{SSS}^{\mathit{PUB}}$, $\mathit{unlinkable}\mathcal{SSS}$, and $\mathit{pubacc}\mathcal{RSS}$ achieve mandatory and all recommended optional parts of Requirement 4 because they are publicly non-interactive accountable (PUB) and achieve third-party Signer-, as well as Sanitizer-accountability.
Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate)	✓	17.3.5	All proposed schemes fulfil the requirement because they facilitate asymmetric cryptographic signature schemes and enable to re-use legally accepted certification of public keys belonging to legal signatory.
Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD))	✓	17.3.6	$\mathit{pubacc}\mathcal{SSS}^{\mathit{Block}}$ (see Sec. 13.10.2) and $\mathit{merge}\mathcal{RSS}$ (see Sec. 14.15) run on a standard smart card.
Requirement 7 (Mechanism MUST achieve a legally accepted state-of-practice strength of the cryptographic algorithm used for integrity and authenticity protection)	✓	17.3.7	$\mathit{pubacc}\mathcal{SSS}$ and $\mathit{pubacc}\mathcal{SSS}^{\mathit{Block}}$ generate signatures using a legally accepted standard hash function and signature scheme.

Table 28. Mandatory requirements 1-7 for increased probative value and their fulfilment by proposed schemes; namely $\mathit{pubacc}\mathcal{SSS}^{\mathit{Block}}$ and $\mathit{pubacc}\mathcal{RSS}$ achieve all mandatory and the recommended, but optional, requirements

17.3.1. Fulfilment of Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer): Integrity is protected to the signatory's policy

Requirement 1[914] limits the subsequent modifications that still yield a valid signature verification to those being authorized. If the MSS scheme is considered secure it especially fulfils the properties of unforgeability and immutability. Thus an adversary is unable to subsequently modify the integrity protected message in an unauthorized way without being detected by a failure of the signature verification. The properties have been defined for SSS and RSS in a harmonised notation based on the existing body of work[915]. There was no need to refine those properties as the definitions in the state of the art offered no shortcomings. Merely the fact that many state-of-the-art RSS schemes did not allow to specify a policy made it unnecessary to discuss immutability for RSS. Still, this thesis added a definition of immutability for RSS following the ideas of immutability for SSS (see Sec. 11.13.2).

[914] Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer) (see Sec. 7.1).

[915] Unforgeability for SSS (see Sec. 11.6.1), immutability for SSS (see Sec. 11.6.2), unforgeability for RSS (see Sec. 11.13.1), and immutability for RSS (see Sec. 11.13.2).

All eight proposed schemes (and all ten constructions) fulfil Requirement 1. The schemes fulfil Requirement 1 if the analysis assumes the following: The scheme's limit(s) in its protective scope are known to the signatory. Thus the scheme being selected means that the signatory consents to their limits, i.e. they are deemed as suitable for the signatory's intended protection. An overview is provided in Tab. 29.

Proposed Scheme (Data Structure)	Requirement 1: Protect content-, structural-, and referential Integrity		Proof of Immutability or Unforgeability	Publication
$pubacc\mathcal{SSS}$ (list)	✓	proven by *Brzuska et al.* [65]	Sec. 13.5.3	[68]
$blockacc\mathcal{SSS}$ (list)	✓	immutable and unforgeable	Sec. 13.7.4	[132]
$unlinkable\mathcal{SSS}$ (list)	✓	immutable and unforgeable	Sec. 13.9.4	[69]
$priv\mathcal{RSS}$ (tree)	✓	unforgeable	Sec. 29	[425]
$flex\mathcal{RSS}$ (tree)	✓	unforgeable	Sec. 14.8.4	[134], [394]
$struct\mathcal{RSS}$ (list)	✓	unforgeable	Sec. 14.10.4	[426]
$merge\mathcal{RSS}$ (set)	✓	unforgeable	Sec. 14.12.4	[393], [387]
$pubacc\mathcal{RSS}$ (list)	✓	unforgeable and Sanitizer-unforgeable	Sec. 14.14.3	[388]

Table 29. All proposed schemes (and all their constructions) from this thesis fulfil Requirement 1

17.3.2. Fulfilment of Requirement 2: Signatory's consent is verifiable

Requirement 2[916] seeks to make the signatory accountable for the integrity protection achieved through the MSS. For all the proposed RSS and SSS of this thesis one must assume that the signatory is fully aware of the protection offered by each scheme and is able to correctly define the documents blocks and define the correct policy to authorize only those subsequent modifications that the signatory intends to authorize.

In detail, the Signer can exercise control over the scope and strength of the integrity and authenticity protection and by generating a signature with a chosen MSS mechanism changelessly fixes all of the required information from Requirement 2.1[917] as identified in Tab. 30: (1) which blocks of an integrity-protected message are possibly modifiable, (2) which modifications are authorized, (3) the level of detection of modifications, and (4) the level of accountability.

Following Requirement 2.2[918] the signatory must be made to fix the scope and document its consent. All schemes clearly state their protective scope and require the signatory to explicitly specify what can be redacted and control the consequences with different flexibility. For example $priv\mathcal{RSS}$ does require to provide explicit new locations for the subtrees if a non-leaf node is redactable. By this the protective scope of each signature is known to the signatory.

The chosen mechanism and its configuration constitute the integrity policy which is signed and thus verifiably consented to by the signatory. This thesis assumes a scheme being selected means that the signatory consents to its limits, i.e. the limitations are deemed as suitable for the signatory's intended protection. As the signature's value also depends on the scheme being used, the valid signature verification outcome transports that that specific scheme was used in consent (see also the discussion offered for Analysis Result 8[a]).

[a] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

[916] Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier) (see Sec. 7.2).

[917] Requirement 2.1 (Mechanism MUST enable the Signer to control scope and strength of the integrity and authenticity protection) (see Sec. 7.2.1).

[918] Requirement 2.2 (Mechanism MUST verifiably document Signer's consent for chosen integrity and authenticity protection) (see Sec. 7.2.2).

Lastly, only the optional Requirement 2.3[919] states that the verifiability of that consent SHOULD be non-interactive. This is given if the scheme or its construction exhibits PUB accountability. Tab. 30 on page 464 gives an overview of the overall fulfilment of all three sub-requirements regarding the signatory's consent.

Proposed Construction (Data Structure)	Which blocks are modifiable (ADM)?	Which modifications are authorized?	Level of detection?	Level of accountability?	Fulfil Req. 2?	Fulfil Req. 3?
pubacc$\mathcal{SSS}$ (list)	each list element	sanitization	1CD	PUB	✓	✓
pubacc$\mathcal{SSS}^{Block}$ (list)	each list element	sanitization	BCD	PUB	✓	✓
blockacc$\mathcal{SSS}^{INT}$ (list)	each list element	sanitization	≥1CD , <BCD	INT	✓£	✓
blockacc$\mathcal{SSS}^{PUB}$ (list)	each list element	sanitization	≥1CD , <BCD	PUB	✓	✓
unlinkable$\mathcal{SSS}$ (list)	each list element	sanitizations	1CD	PUB	✓	✓
priv$\mathcal{RSS}$ (tree)	intermediate and leaf-node	redaction followed by relocation of subtree within transitive closure of tree as allowed by Signer	UCD	INT	✓£	$
flex$\mathcal{RSS}$ (tree)	intermediate and leaf-node	redaction followed by relocation of subtree to any tree location as allowed by Signer	UCD	INT	✓£	$
struct$\mathcal{RSS}$ (list)	each list element + relation between them	redaction of element's content or redaction of relation towards other elements (order)	UCD	INT	✓£	$
merge$\mathcal{RSS}$ (set)	each set element	redaction	UCD	INT	✓£	$
pubacc$\mathcal{RSS}$ (list)	each list element	redaction	1CD	PUB	✓	✓

Table 30. All proposed constructions fulfil the mandatory parts of Requirement 2; they differ in protection scope and in their flexibility to define the policy of authorized subsequent modifications (ADM); constructions marked with £ do not achieve the recommended but optional Requirement 2.3; only constructions with ≥1CD fulfil Requirement 3; constructions marked with $ require additional methods like ARSS to achieve 1CD (see Theorem 44 on page 422) as they only achieve UCD

All ten proposed constructions fulfil Requirement 2. However, not all constructions achieve the optional Requirement 2.3 which recommends that the signatory's consent can be publicly verified. This is only possible for constructions that offer non-interactive public accountability (PUB). In Tab. 30 this is indicated by putting a £-sign next to the checkmark, this means the construction only fulfils the mandatory parts from Requirement 2.

[919] Requirement 2.3 (Mechanism SHOULD enable Verifier to verify the proof of consent non-interactively and publicly) (see Sec. 7.2.3).

17.3.3. Fulfilment of Requirement 3: Any subsequent modification is detected ($\geq$1CD), either INT or PUB

Requirement 3[920] is split into three sub parts; each is analysed in turn. Requirement 3.1[921] requires that the mechanism MUST verify the absence of **any** subsequent modifications. This is covering any subsequent modification and not just unauthorized. Hence, ACA – UCD integrity is not enough for MSS[922]. Hence, the proposed MSS mechanism are analysed for their achievement of $\geq$1CD integrity. The following schemes inherently achieve this: $pubacc\mathcal{SSS}$, $pubacc\mathcal{SSS}^{Block}$, $blockacc\mathcal{SSS}^{PUB}$, $unlinkable\mathcal{SSS}$, and $pubacc\mathcal{RSS}$. This is also shown in Tab. 30 which already lists all proposed constructions' level of detection. Constructions marked with a \$-sign in Tab. 30 do inherently only achieve ACA – UCD integrity but can be combined with an SSS to achieve ACA – $\geq$1CD –PUB using the ARSS framework (see Sec. 14.1).

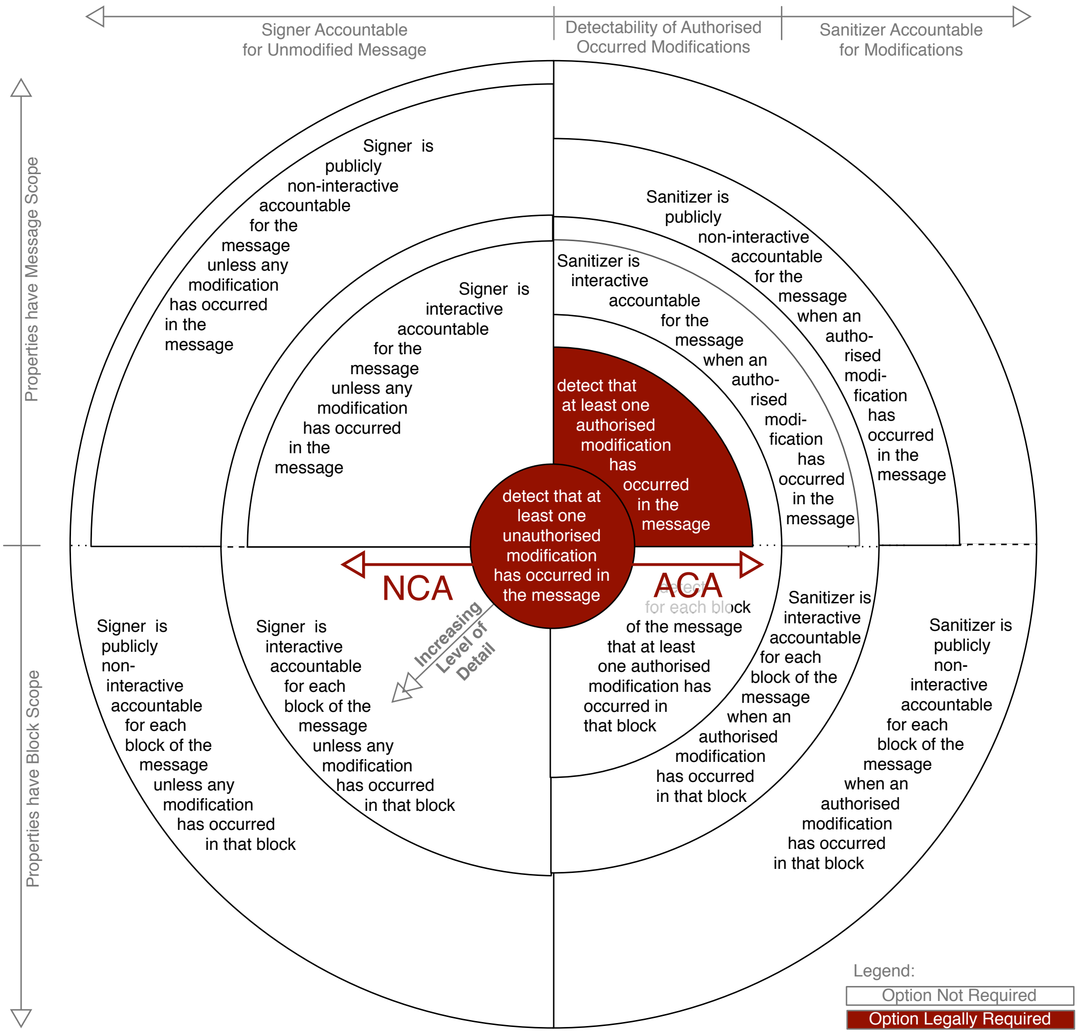

Signer Accountable
for Unmodified Message
Detectability of Authorised
Occurred Modifications
Sanitizer Accountable
for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an authorised modification has occurred in the message
Sanitizer is interactive accountable for the message when an authorised modification has occurred in the message
detect that at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
NCA
ACA
Increasing Level of Detail
detect for each block of the message that at least one authorised modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Legend:
Option Not Required
Option Legally Required

Figure 130. Legally required integrity protection for mechanisms in which authorized subsequent changes are allowed (ACA integrity) in fulfilment of Requirement 3: If one or more subsequent modification (authorized or unauthorized) has occurred, this must be detected as modified.

[920] Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification) (see Sec. 7.3).

[921] Requirement 3.1 (Mechanism MUST verify the absence of any subsequent modifications, i.e., $\geq$1CD integrity) (see Sec. 7.3.1).

[922] Of course, if no changes are authorized (NCA) then NCA – $\geq$UCD is enough. But this thesis is concerned with MSS that allow for ACA instead of NCA.

Requirement 3.2[923] requires that the mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory. All proposed schemes build on asymmetric credentials for Signer and Sanitizer, i.e., public keys like $\mathsf{pk}_{\mathsf{san}}$ and $\mathsf{pk}_{\mathsf{sig}}$ exist. Those public keys can be put under the government of legally accepted PKI and by this flanking technical and organisational measures linked to a legal signatory.

Finally, Requirement 3 contains a recommendation: Requirement 3.3[924] recommends that the mechanism SHOULD achieve non-interactive public verification of integrity.

The reason for the recommendation comes from comparison of existing legally accepted CDSS. CDSS — while they do not allow authorized changes (NCA = no subsequent changes allowed) — also require that unauthorized subsequent modifications are detected, which in NCA automatically means that all subsequent modifications must be detected (NCA – UCD = NCA – 1CD). For an RSS or SSS the detection of allowed authorized modifications (ACA = authorized subsequent changes allowed) is handled separately: If at least one unauthorized occurred change is detected by the scheme (ACA – UCD), then authorized subsequent modifications might not be correctly detected with overwhelming probability and thus this scheme would not detect that at least one occurred change has happened (1CD). Hence, ACA – UCD < ACA – 1CD = NCA – 1CD.

Thus, RSS or SSS must detect

- when at least one unauthorized modification, **as well as**
- when at least one authorized modification

has occurred in the message.

This integrity protection need is depicted in Fig. 130. Within the visual representation used throughout the thesis this results in a colouring of the inner circle — the 'bull's eye' — for the detection of all unauthorized modifications. This would be enough for NCA protection mechanism. ACA schemes additionally require to detect also any authorized modification (≥1CD) which is depicted by having the first sector of the upper right quadrant coloured.

The constructions $pubacc\mathcal{SSS}$, $pubacc\mathcal{SSS}^{Block}$, $blockacc\mathcal{SSS}^{PUB}$, $unlinkable\mathcal{SSS}$, and $pubacc\mathcal{RSS}$ achieve the mandatory and the recommended parts of Requirement 3. It is mandatory to achieve ACA – ≥1CD – INT, as achieved by the transparent construction $blockacc\mathcal{SSS}^{INT}$. Those four mechanisms were designed to achieve the recommended ACA – ≥1CD – PUB integrity. They exhibit a reduced functional gap towards a CDSS which achieves NCA – 1CD – PUB (see Sec. 6.10).

The constructions $priv\mathcal{RSS}$, $flex\mathcal{RSS}$, $struct\mathcal{RSS}$, do not achieve ≥1CD on their own; If they are combined with an ≥1CD – PUB SSS (following the ARSS framework) they achieve Requirement 3. The redactable signature schemes that achieve ACA – ≥UCD but not the required or recommended accountability can be paired with a sanitizable signature scheme following the ARSS framework. Using the $pubacc\mathcal{RSS}$ scheme shown in Sec. 14.14.1 each of the schemes can fully keep their flexibility gains and achieve ACA – ≥1CD – PUB in combination with any secure SSS that achieves this (see Theorem 44 on page 422).

17.3.4. Fulfilment of Requirement 4: Based on detection of any occurred subsequent modification (≥1CD) achieve Signer accountability

Further to the detectability of modifications the accountability might need to be transferred away from the Signer after a subsequent modification. Obviously, if an **un**authorized change has happened and is detected, neither Signer nor Sanitizer are accountable. Requirement 4 is concerned with the accountability, especially that of the Signer. The requirement is split, Requirement 4.1[925] is mandatory, while

[923] Requirement 3.2 (Mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory) (see Sec. 7.3.2).

[924] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) (see Sec. 7.3.3).

[925] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1).

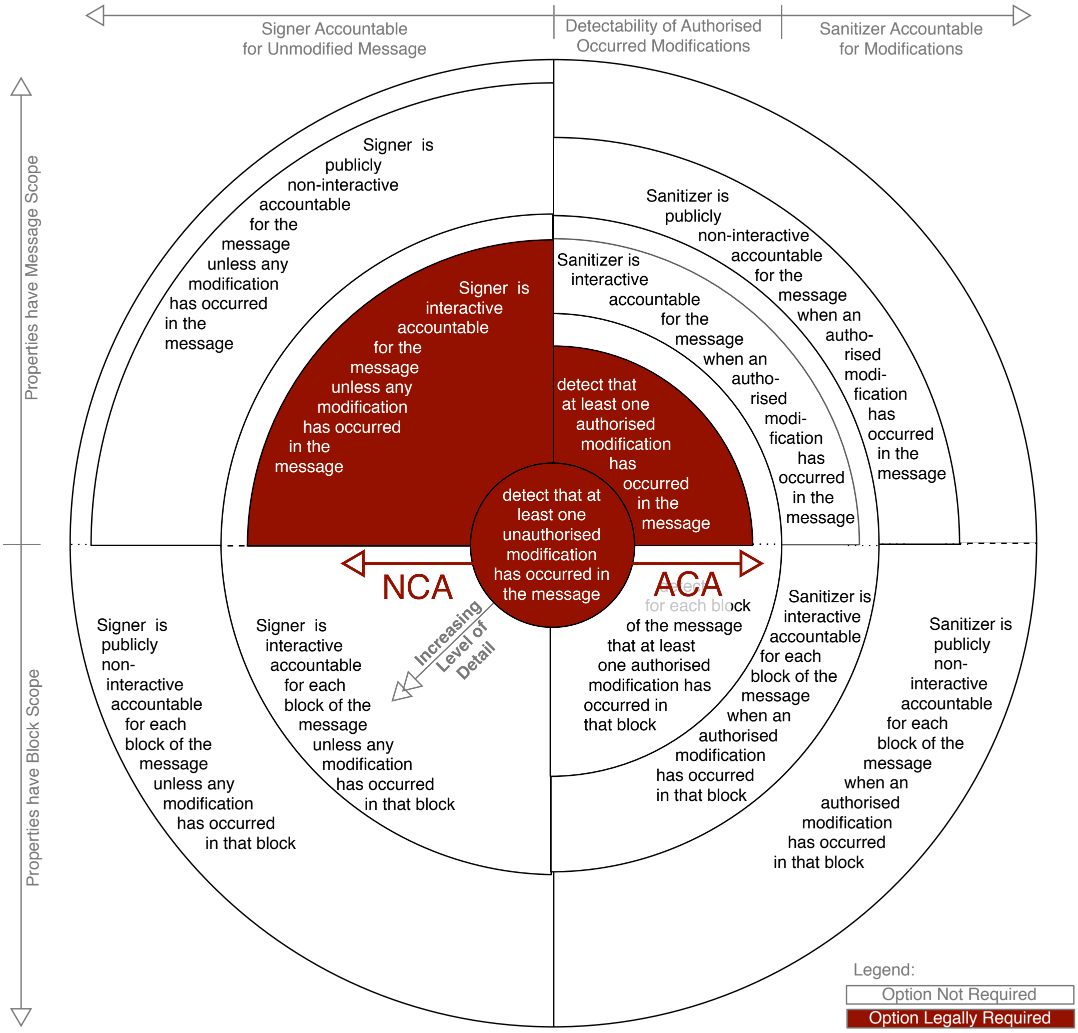

Signer Accountable for Unmodified Message
Detectability of Authorised Occurred Modifications
Sanitizer Accountable for Modifications
Properties have Message Scope
Properties have Block Scope
Signer is publicly non-interactive accountable for the message unless any modification has occurred in the message
Signer is interactive accountable for the message unless any modification has occurred in the message
Sanitizer is publicly non-interactive accountable for the message when an autho-rised modi-fication has occurred in the message
Sanitizer is interactive accountable for the message when an autho-rised modi-fication has occurred in the message
detect at least one authorised modification has occurred in the message
detect that at least one unauthorised modification has occurred in the message
NCA
ACA
detect for each block of the message that at least one authorised modification has occurred in that block
Increasing Level of Detail
Signer is publicly non-interactive accountable for each block of the message unless any modification has occurred in that block
Signer is interactive accountable for each block of the message unless any modification has occurred in that block
Sanitizer is interactive accountable for each block of the message when an authorised modification has occurred in that block
Sanitizer is publicly non-interactive accountable for each block of the message when an authorised modification has occurred in that block
Legend:
Option Not Required
Option Legally Required

Figure 131. Legally required integrity protection according to the mandatory parts of Requirement 3 and Requirement 4 for mechanisms in which authorized subsequent changes are allowed (ACA integrity)

Requirement 4.2[926] is optional. Requirement 4.1 contains a mandatory clause in order to require that the mechanism MUST achieve accountability of the Signer including achieving technical non-repudiation of signature creation. It further recommends that the scheme SHOULD achieve non-interactive public accountability of the Signer. Note, compliance with the requirement of accountability is closely linked to the detectability. Legally, the electronic signature needs to be "[...] linked to the data to which it relates in such a manner that any subsequent change of the data is detectable [...]" [Art. 36 (d), Regulation 910/2014]. This dictates that any subsequent modification MUST be detectable (see Sec. 17.3.3). Moreover in RSS and SSS the detection of modifications or the detection of their absence MUST also yield a verification outcome that for the correct cases hold the Signer accountable. A third-party verifiable proof of accountability of the Signer means that the entity in the role of the Signer is technically accountable for having generated the message with its current contents.

This integrity protection need is depicted in Fig. 131. Within the visual representation used throughout the thesis this results in a colouring of the inner circle — the 'bull's eye' — for the detection of all unauthorized modifications. For ACA schemes additionally the first sector of the upper right quadrant is marked, due to the need to detect also any authorized modification. So far this is the same as the require-

[926] Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4.2).

ments for Requirement 3 previously depicted in Fig. 130. Due to the Signer accountability required by Requirement 4 the first sector of the upper left quadrant is also coloured Fig. 131. For NCA schemes the upper right corner is not required as you can see if you visually compare it to the visualisation of a CDSS as given in Sec. 6.9 on page 159.

Requirement 4.1 further recommends that the scheme SHOULD achieve non-interactive public accountability (PUB) of the Signer. It is clear from the legal texts that a digital signature scheme needs to provide ≥1CD detection. However, achieving this appears to pose a legal problem when an MSS is specifically used to make it possible for a Sanitizer to alter the message after it has been signed without such a change being obviously detectable for a Verifier. This is the case if the scheme gives accountability but only offers interactive non-public accountability (INT), e.g. transparent schemes. *Roßnagel* in [410] — citing [147, 149] — states that the identification function is no longer provided if the entity generating the signature is not the entity that controls the signature generation key, i.e., the Signer. [927] Having the Sanitizer non-interactive publicly accountable for any occurred and authorized subsequent modification allows the Verifier to appoint the Sanitizer as the origin, which weakens the argument that the identity function cannot be fulfilled. Achieving non-interactive public accountability (PUB) provides an option to publicly distinguish between a Signer generated signature and a subsequently modified one[928].

First and foremost, the fulfilment of Analysis Result 4 enables the Verifier to identify the presence or absence of at least one occurred subsequent modification from just the Signer's public verification key, the message and the signature. Given this, it is 'easy' or 'simple' for the Verifier to carry out the identity function[929]. Further, it enables the expected authenticity functionality, which allows the Verifier to assume that the content of the signed document originated from the signatory[930]. More details of this argumentation were given in Sec. 7.4.1.

The constructions $pubacc\mathcal{SSS}$, $pubacc\mathcal{SSS}^{Block}$, $blockacc\mathcal{SSS}^{PUB}$, $unlinkable\mathcal{SSS}$, **and** $pubacc\mathcal{RSS}$ **achieve the mandatory parts of Requirement 4.** They provide ≥1CD and Signer-accountability.

The second sub-requirement Requirement 4.2[931] is optional. It requests that after an authorized modification the Sanitizer becomes accountable. Accountability of the Sanitizer can be used to attach a legal probative value to the sanitized document as offered by existing CDSS affixed by the Sanitizer. This is discussed in Sec. 17.5.2.4 (3^{rd} case of a verification).

The constructions $pubacc\mathcal{SSS}$, $pubacc\mathcal{SSS}^{Block}$, $blockacc\mathcal{SSS}^{PUB}$, $unlinkable\mathcal{SSS}$, **and** $pubacc\mathcal{RSS}$ **achieve the mandatory and all recommended optional parts of Requirement 4.** This includes the recommended optional part of Requirement 4.4 to be publicly non-interactive accountable (PUB) and the optional part of achieving third-party Sanitizer-accountability (Requirement 4.2). The five listed constructions provide ≥1CD – PUB and Signer-, as well as Sanitizer-accountability.

17.3.5. Fulfilment of Requirement 5: Linkage towards the legal signatory

This requirement can not be achieved by the cryptographic signature algorithm alone, but entails additional layers. To achieve Requirement 5[932] the current CDSSs build upon asymmetric cryptography and the identification function is achieved by relying on public key infrastructures (PKI). Still, there remains one central technical demand for the actual cryptographic mechanism: The mechanism needs to base on asymmetric cryptography, only then the signature generation key (sk_{sig}) can be kept secret and under the Signer's control and the verification key (pk_{sig}) can be made public.

[927] In German: "Sie [die Identitätsfunktion] ist jedoch nicht mehr gewährleistet, wenn "die Person des tatsächlich Signierenden nicht mit der des Signaturschlüsselinhabers überein(stimmt)" [147, 149]" [410].

[928] The inability to distinguish the 'remote signature' of Regulation 910/2014 from a directly Signer-generated one has been criticised by *Roßnagel* [410].

[929] In German BT-Drs 14-4987 states "[...] kann der Adressat in **einfacher Weise** überprüfen, ob die Signatur **mit dem privaten Signaturschlüssel des Schlüssel-Inhabers erstellt** worden ist und ob der signierte Text **nachträglich verändert worden** ist." [147]; bold face has been added for highlighting.

[930] In German it states "Hierdurch soll gewährleistet werden, dass die Erklärung **inhaltlich vom Unterzeichner herrührt**." [147]; bold face has been added for highlighting.

[931] Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4.2).

[932] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate) (see Sec. 7.5).

To achieve the linkage — that are currently already in place — can be facilitated. This means that whenever an MSS can be instantiated with a signature algorithm for the Signer 's signature that is legally established and for which the keys are hence compatible with the already deployed public key certificates, the existing trust infrastructure can be immediately facilitated to achieve the linkage and thus fulfil Requirement 5.

This thesis moved the key management and the provision of such linkage infrastructures to be out of scope (see Sec. 1.4.3). While it is not an easy task to operate this infrastructure [220] it still is "[...] simply too valuable than not to be used in practice." [320]. MSS mechanisms that can work upon the existing infrastructure offer a lower adoption barrier as these infrastructures are already the 'fabric' of current legally accepted signatures [170, 193].

All eight proposed schemes fulfil Requirement 5. The proposed schemes can do their share to fulfil Requirement 5 (a) if the scheme is based on asymmetric keys and (b) if the signature scheme selected as one of the building blocks achieves Requirement 5, e.g. using RSASSA-PSS from PKCS-v2.2 [422] as the signature algorithm. Especially, the mechanisms $pubacc\mathcal{SSS}$ from Sec. 13.4 can be instantiated using RSASSA-PSS algorithms [422] (see Sec. 13.5). If this same SSS is also used in the ARSS framework the resulting $pubacc\mathcal{RSS}$ is also build using RSASSA-PSS algorithms and thus can achieve Requirement 5.

17.3.6. Fulfilment of Requirement 6: Run secret-key-relevant operations inside QSCD/SSCD

Requirement 6[933] demands that the operations that require access to secret keys can be run inside a confined and secure environment. For legally recognised signatures this is currently realised by using hardware security modules (HSM) as defined in Sec. 2.13. The idea is that the secret signature generation data never leaves the HSM, e.g., a secure smart card. The HSM then offers additional layers of protection against adversaries trying to extract information about the key. The prototypical realisations of several proposed and existing schemes on a secure smart card[934] show that Requirement 6 can be accomplished for MSS.

$pubacc\mathcal{SSS}^{Block}$ and an adapted $merge\mathcal{RSS}$ runs on a standard of the shelf secure smart card. The implementation of $pubacc\mathcal{SSS}^{Block}$ shows that a non-interactive publicly accountable scheme which allows to determine the origin of each block (see Sec. 13.10.2). Also $merge\mathcal{RSS}$ was shown to run on a standard smart card after an adaption (see Sec. 14.15). Moreover, for all prototypes a commercially available secure smart card was used to show that no special hardware or software is needed.

17.3.7. Fulfilment of Requirement 7: Provably secure algorithms

The proposed schemes are built on top of existing primitives. For example $pubacc\mathcal{SSS}$ (see Construction 1 on page 333) is secure if the underlying signature scheme S is UNF-CMA (see Theorem 13 on page 336). Then it can be instantiated with schemes like RSASSA-PSS [39].

Of course an algorithm would not be legally strong if an adversary can create a forgery with 'relatively justified effort'[935] [409]. To the best of the author's knowledge the provable security[936] achieved by the formal security proofs for the schemes presented in this thesis have not been broken at the time of writing[937]. The current level of scrutiny was the peer-review of academic conferences and a journal. This might further increase if the schemes become internationally standardised or practically used. In the case of MSS this thesis thinks that the constructions presented within this thesis will stand the current challenges and the achieved strength can be backed by an expert opinion. *Grigorjew* [219] explains that

[933] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

[934] The smart card was a "SmartC@fé® Expert 4.x" from *Giesecke and Devrient* [206].

[935] Translated from the German "verhältnismäßigem Aufwand" [409, p. 166].

[936] Provable security is defined and explained in Sec. 2.3.1.

[937] Even against provably secure schemes it sometimes takes time until attacks are found and published (cmp. [492]), or it needs time until the algorithms have gained enough usage for more cryptographers to show interest in breaking them.

the bench[938] will judge the algorithm's strength and suitability at the court's discretion and conviction[939] and thus will, additionally to the technical expert opinion, base the final decision on the motives and the availability of the needed cryptographic and technical skills of the accused [219, p. 215–216]. Generally, an objection against the strength of an algorithm is made harder if the cryptographic building blocks from which the MSS is constructed are legally already accepted as strong enough. In this thesis many of the constructions can be instantiated from signature schemes that are legally accredited, e.g., $\mathit{pubacc}\mathcal{SSS}$ and $\mathit{pubacc}\mathcal{SSS}^{\mathit{Block}}$ they can be based upon an UNF-CMA signature scheme without needing additional hash-functions. As such the objection would result in an expert opinion which if following the arguments regarding the strength of the algorithm given in this thesis would indicate a sufficient strength of $\mathit{pubacc}\mathcal{SSS}$ or $\mathit{pubacc}\mathcal{SSS}^{\mathit{Block}}$ instantiated upon RSASSA-PSS from PKCS-v2.2 [422].

$\mathit{pubacc}\mathcal{SSS}^{\mathit{Block}}$**,** $\mathit{pubacc}\mathcal{SSS}$ **are proven to inherit their security from legally accepted standard hash functions and signature schemes and thus fulfil Requirement 7.** The construction of $\mathit{pubacc}\mathcal{SSS}$ was proven to be secure if the underlying signature scheme S is UNF-CMA (see Theorem 13 on page 336) which can be provided by a mechanism with legally accepted strength, i.e. RSASSA-PSS from PKCS-v2.2 [422] is registered in the catalogue of schemes with currently acceptable strength (see Sec. 5.3.1).

17.4. Evaluation Result 2: MSS are 'elementary' electronic signatures following Regulation 910/2014

First and foremost, this section will show that MSS generates a signature to meet the minimal legal requirements according to Regulation 910/2014: *electronic signatures*. To strengthen that this is just the minimum the term 'elementary' is added even-though this is not a legal term. The goal is to become recognised as a means to generate *qualified electronic signatures* according to Regulation 910/2014. The following Tab. 31 lists the requirements that are defined in Article 3 section 10 Regulation 910/2014 and the reason why the thesis evaluates that MSS are electronic signatures following Regulation 910/2014[940]. First, the signature value σ created by an MSS is (1) in electronic form, it can thus be attached to the document that was signed — however that link is not so important for electronic signatures. Second, due to the cryptographic property of unforgeability and the use of asymmetric cryptography the signature is also (2) logically associated with (3) the signatory's secret signature generation key ($\mathsf{sk}_{\mathrm{sig}}$). As such this thesis concludes — quite obviously — the following:

Interpreted[940] **Requirements from Art. 3 (10) Regulation 910/2014**	**MSS**	**Note**
(1) *signature* needs to be in electronic form;	✓	signature σ itself is in digital form
(2) *created signature* needs to be attached to, or logically associated with, other electronic data, *i.e., the* $\mathsf{sk}_{\mathit{sig}}$	✓	asymmetric cryptography and cryptographic unforgeability allow the MSS to bind the electronic data m, the signature σ, and the keys $\mathsf{sk}_{\mathrm{sig}}$ and $\mathsf{pk}_{\mathrm{sig}}$
(3) *other electronic data, i.e. the* $\mathsf{sk}_{\mathit{sig}}$, needs to be used by the signatory to sign.	✓	

Table 31. All schemes proposed in this thesis are electronic signatures following Regulation 910/2014; *italics* it used to identify interpreted wording, original in Footnote 940.

A signature created by an MSS that fulfils the mandatory requirements 1–7 is an **electronic signature** according to Article 3 Section 10 Regulation 910/2014.

However, a signature that is just an electronic signature does not induce any legal effects regarding a statutory right to an increased probative value of the document that bears such an electronic signature. An electronic signature is the 'elementary' level of electronic signatures which does allow for a legal recognition, i.e., it "[...] shall not be denied legal effect and admissibility as evidence in legal proceedings solely on the grounds that it is in an electronic form [...]" [Art. 25 Regulation 910/2014]. To increase the legal probative value the signature created by the MSS the scheme needs to achieve the level of a

[938] 'Bench' here is the legal term referring to the judge(s) taking the decision in court during a legal case.

[939] In German "Freie Beweiswürdigung" [99].

[940] Original text: "'electronic signature' means data in electronic form which is attached to or logically associated with other data in electronic form and which is used by the signatory to sign;" [Art. 3 (10) Regulation 910/2014].

qualified electronic signature as they "[...] have the equivalent legal effect of a hand-written signature [...]" [Art. 25 Regulation 910/2014]. Only for qualified electronic signatures there exists legal texts assigning statutory evidentiary presumption to the signed document. And only then the document receives a high probative value[941]. The mandatory parts of the requirements 1–7 created in this thesis are exactly due to Regulation 910/2014 and the national signature legislations which demand additional qualities from a qualified electronic signature compared to the 'elementary' electronic signature.

17.5. Comparison of signature verification outcomes of an MSS and a qualified electronic signature generated by a CDSS

In the previous Sec. 17.3, the proposed schemes were evaluated against each of the requirements for an increase of the probative value. Following that evaluation two mechanisms are selected because they achieve all mandatory technical requirements and additionally the optional ones that are recommended to close the gap between legally accepted electronic signature mechanisms and MSS:

$pubacc\mathcal{SSS}$ (see Sec. 13.4.1), and $pubacc\mathcal{RSS}$ (see Sec. 14.13.1).

Each of the two chosen MSS achieves the following:

- The MSS achieves the mandatory prerequisites from Requirements 1–7.
- The MSS achieves Requirement 4.1[942] to the full extent[943] that recommends that the mechanism SHOULD achieve non-interactive public accountability (PUB) of the Signer.
- The MSS achieves Requirement 3.3[944] that recommends that the mechanism SHOULD achieve non-interactive public verification of integrity.

The following evaluation compares schemes with those properties to a legally accepted CDSS. Due to their alignment with the requirements the similarities are big and the differences are kept small. As a result there is finally only one remaining technical difference in the integrity protection: MSS (like $pubacc\mathcal{SSS}$ or $pubacc\mathcal{RSS}$) allow to authorize subsequent changes (ACA) while a CDSS does not (NCA).

Sec. 17.5.1 will briefly state the two cases distinguished by the outcome of a validation of a CDSS-based signature. This is compared with the cases — five cases in total — introduced by MSS with ACA – ≥1CD – PUB integrity. If the cases of the MSS achieve an equal statement to that made for a case generated by the CDSS generated than this thesis sees them as providing an increased probative value. For each case the analysis indicates the probative value that could be awarded to such a document.

17.5.1. Two cases after verifying a CDSS with NCA – 1CD integrity

When verifying the document m and the signature σ generated by a legally accepted CDSS that does not allow authorized modifications (NCA) but offers ≥1CD detection using the signatory's public signature verification key $\mathsf{pk}_{\mathrm{sig}}$ the following two cases can be distinguished:

Case 0: Signature σ is **invalid** under $\mathsf{pk}_{\mathrm{sig}}$.

Case 1: Signature σ verifies under $\mathsf{pk}_{\mathrm{sig}}$ as **valid**, and

≥1CD detection yields that **no subsequent** modification has happened.

This is depicted in Fig. 132 in order to compare this with the visualisation for MSS given in Fig. 133.

[941] For example the German laws, previous to eIDAS, stated that only if the electronic document "[...] bear[s] a qualified electronic signature in accordance with the Electronic Signature Act [...]" [§ 87a paragraph 3 AO] the electronic form becomes a subsidiary to the paper form.

[942] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1).

[943] To the full extent meaning that additionally to the mandatory part also the recommended part is achieved.

[944] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) (see Sec. 7.3.3).

17.5.1.1. Case 0: Invalid signature

With the signature being invalid, no increase of probative value can be deducted. The reason may vary, but this thesis found none where an increased probative value can be assigned. An invalid signature might even decrease the probative value compared to an unsigned document, however this was not further analysed as it is considered out of scope.

17.5.1.2. Case 1: Valid signature and detection of occurred subsequent modifications yields document is original and unmodified

In Case 1 a verifiable proof of the Signer being the authentic origin of the document, which has seen no subsequent modifications. Following legal practice and prevailing opinions of legal theorists this thesis reached this statement by assuming that the CDSS fulfils all seven requirements, especially the mechanism MUST achieve state-of-the-art unforgeability (Requirement 7[945]) and is generated on a QSCD (Requirement 6[946]) and makes use of legally qualified certificates (Requirement 5[947]). Especially, the signatures generated are legally recognised as equivalent to handwritten wet-ink signatures with a high legal certainty under the assumption that the electronic signature scheme's strength is all of the following: (1) cryptographically considered strong (Requirement 7.1[948]), and (2) as strong as those mechanisms legally accepted (Requirement 7.2[949]), and (3) legally recognised (Requirement 7.3[950]).

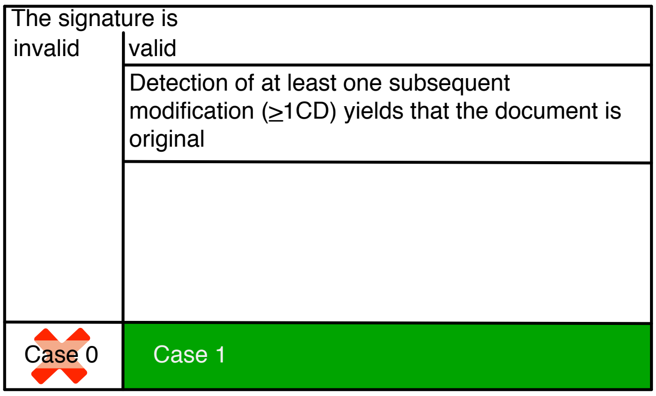

The signature is
invalid
valid
Detection of at least one subsequent modification (≥1CD) yields that the document is original
Case 0
Case 1

Figure 132. Cases of CDSS signature verification results; styled to be visually comparable to Fig. 133.

If the mandatory prerequisites from Requirements 1–7 are fulfilled and especially Requirement 7.3, a signed document with a signature verification outcome in case 1 would give the document an increased probative value: prima facie evidence or even evidence with a legal presumption of authenticity following Regulation 910/2014 and ZPO n.F..

[945] Requirement 7 (Mechanism MUST achieve a legally accepted state-of-practice strength of the cryptographic algorithm used for integrity and authenticity protection) (see Sec. 7.7).

[946] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

[947] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate) (see Sec. 7.5).

[948] Requirement 7.1 (Mechanism MUST achieve cryptographic protection against a forgery) (see Sec. 7.7.1).

[949] Requirement 7.2 (Mechanism MUST be as good as the legally accepted current state of practice for integrity and authenticity protection) (see Sec. 7.7.2).

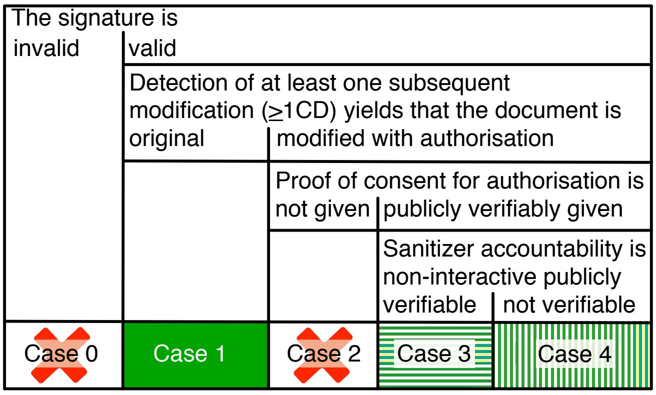
[950] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms) (see Sec. 7.7.3).

Suitable CDSS schemes under Regulation 910/2014 are for example RSASSA-PSS from PKCS v2.2 [422] (listed in [75, 85]). As a secure signature creation device (Requirement 6[951]) a secure smart card enlisted[952] by the national bodies of the EU member states following Article 30 (2) and 39 (2) of Regulation 910/2014 can be used. Obviously the schemes are based on asymmetric cryptography, as such the public key can be certified by an accredited certification authority, such as those listed in [170] (Requirement 5[953]). In combination this means that the signed document offers an increased probative value over an unsigned document.

Note, this 'wet-ink equivalence' only holds as long as the law does not explicitly demand a handwritten document[954].

17.5.2. Five distinguishable results after verifying an MSS with ACA – ≥1CD integrity and public 3rd-party verifiability (PUB)

Compared to a CDSS there are additional cases when verifying a document m and a signature σ generated by an RSS or an SSS that offers ≥1CD integrity detection. The assumption is that one is using the signatory's public signature verification key $\mathsf{pk}_{\mathrm{sig}}$ for verification. The evaluation also assumes that the MSS offers the public 3rd-party verifiability (PUB), as do CDSS. Note, Case 0 and Case 1 of the MSS correspond to the respective case of a CDSS because this analysis is for RSS or SSS that as well offer ≥1CD – PUB detection. Case 2 is mentioned to explicitly highlight that flanking technical and organisational mechanisms are even more important for MSSs; they are also important for CDSS because if the signatory's consent to the generation of the electronic signature can be cast into serious doubt, then also CDSS have legal problems. Case 3 describes the case when a Sanitizer is accountable for an authorized subsequent modification, while Case 4 occurs if the MSS would only offer Signer-accountability but no Sanitizer accountability. The latter matches for RSS, where the sanitization operation of redaction is considered a public operation.



The signature is
invalid
valid
Detection of at least one subsequent modification (≥1CD) yields that the document is
original
modified with authorisation
Proof of consent for authorisation is
not given
publicly verifiably given
Sanitizer accountability is non-interactive publicly
verifiable
not verifiable
Case 0
Case 1
Case 2
Case 3
Case 4

Figure 133. Cases of MSS properties and signature verification results:
Case 1 resembles a CDSS on an original document by the Signer;
Case 3 resembles a CDSS on the subsequently modified document, which could be considered an original created by the Sanitizer.

Recall[955], a detection level of ≥1CD means that as a minimum the mechanism allows the following: The Verifier detects that either no modification or that at least one modification (authorized or unauthorized) to an integrity-protected message has occurred. The exact number of occurred modifications or where they happened remains invisible to the Verifier. Further, recall[956] that the public form of Signer-accountability that is 3rd-party verifiability, termed non-interactive public accountability (PUB) means

[951] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

[952] The EU compiled a list of accredited devices in [171].

[953] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate) (see Sec. 7.5).

[954] A prominent example in Germany for a document that will not be legally recognised if in electronic form is the will. It requires "[...] a declaration written and signed in his own hand."[§ 2247 (1) BGB].

[955] For details see Sec. 6.3 on page 144 and the related Requirement 3.1 (Mechanism MUST verify the absence of any subsequent modifications, i.e., ≥1CD integrity) in Sec. 7.3.1 on page 171.

[956] For details see Sec. 6.4.3 on page 153 for the full definition.

the mechanism allows for a verifying signed message that has been subject to an authorized subsequent modification to corroborate the correct and active involvement of an authorized participant other than the Signer. And that this is 3^{rd}-party-verifiable and thus does neither require the active involvement of another protocol participant, nor the active involvement of a trusted third party, nor the knowledge of another participant's secret keys.

At least five cases can be distinguished for an MSS, as depicted in Fig. 133. Namely, when verifying a signature generated by an MSS with ACA – ≥1CD – PUB integrity integrated into a framework for electronic signatures, i.e. including additional flanking organisational and technical mechanisms, the following results can happen:

Case 0: Signature σ is **invalid** under $\mathsf{pk}_{\mathrm{sig}}$.

Case 1: Signature σ verifies under $\mathsf{pk}_{\mathrm{sig}}$ as **valid**, and

≥1CD detection yields that **no subsequent** modification has occurred.

Case 2: Signature σ verifies under $\mathsf{pk}_{\mathrm{sig}}$ as **valid**, and

≥1CD detection yields that **at least one subsequent** modification has happened, but

no verifiable proof of consent to authorized modifications by the Signer is presented. Note, the reason(s) for the missing verifiable consent lies outside the scope of the technical integrity protection mechanism of the MSS[957].

Case 3: Signature σ verifies under $\mathsf{pk}_{\mathrm{sig}}$ as **valid**, and

≥1CD detection yields that **at least one subsequent authorized** modification has happened, and

a **verifiable proof of consent** from the Signer having authorized the occurred modifications is presented, and

the Sanitizer's accountability can be proven **non-interactive publicly (PUB)**.

Case 4: Signature σ verifies under $\mathsf{pk}_{\mathrm{sig}}$ as valid, and

≥1CD detection yields that **at least one subsequent authorized** modification has happened, and

a **verifiable proof of consent** from the Signer having authorized the occurred modifications is presented, but

the Sanitizer's accountability can be proven **interactive non-publicly (INT)**.

In the following each case is described and the probative value assigned to the signed document is stated based on the previous given analysis and requirements.

17.5.2.1. Case 0: Invalid signatures

No increase of probative value can be deducted in this case. This is the same case as for CDSS (see Sec. 17.5.1.1) with the same result regarding the assignment of no increased probative value.

17.5.2.2. Case 1: Valid signature and detection of occurred subsequent modifications yields document is original and unmodified

In Case 1, the signature of an MSS signature with ACA – ≥1CD and 3^{rd}-party verifiability serves towards a third party as a verifiable proof of the Signer being the authentic origin of the document, which has seen no subsequent modifications. In the same way that the CDSS allows to publicly identify the origin (the Signer) by offering NCA – 1CD – PUB integrity, an MSS with ACA – ≥1CD – PUB allows to verify that the document is original and from the Signer if no subsequent modifications are identified during signature verification.[958]. This is possible for the MSS because with ≥1CD detection the absence of

[957] See Sec. 17.5.2.3 for a brief discussion.

[958] The technical needs that enforce this behaviour have been captured in the mandatory Requirement 3.2 (Mechanism MUST offer non-interactive public authentication of origin through the use of asymmetric cryptography to allow the origin's public key to be linked to a legal signatory) (in Sec. 7.3.2) and the optional Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) (in Sec. 7.3.3).

any subsequent modification can be established, i.e., if no subsequent modification has happened, the Verifier corroborates the absence of any subsequent modification during verification and thus knows that the document is still the original as originated from the Signer. Furthermore, the requested 3^{rd}-party verifiability allows to use the validation of the signature as evidence towards a third party. The evaluation of probative value assumes that the mandatory prerequisites from Requirements 1–7 are fulfilled by the MSS. Here, it is essential that the mandatory Requirement 3.1 is fulfilled as it ensures that the absence of any subsequent modification is verifiable. Further, the fulfilment of the mandatory requirements ensures that the Signer becomes linked to the legal signatory via organisational and additional flanking technical mechanisms[959]. The close relation between MSS with ACA – ≥NCD – PUB integrity and CDSS's NCA – 1CD – PUB integrity allows emulating the behaviour of CDSS if the Signer's policy of the MSS would inhibit all subsequent modifications.

MSS with ACA – ≥NCD – PUB integrity with all blocks marked as non admissible emulates a CDSS. If MSS achieves all the mandatory elements of Requirements 1–7, then the Signer's control for the authorized subsequent modifications and the non-interactive public accountability offered by ACA – ≥NCD – PUB integrity is strongly enforced and any violation is detectable by the Verifier. With the public form of 3^{rd}-party verifiability PUB, the MSS signature achieves the requirement When any subsequent modification is disallowed, i.e., the MSS signature contains only fixed block(s), then the MSS technically behaves like a CDSS.

Because an MSS is most useful if the signatory has actually authorized subsequent modifications the analysis continues for the case that ADM is not empty. Once subsequent modifications are authorized their non-occurrence is differentiated due the detection capability being ≥1CD. The question to answer is whether or not it requires PUB accountability. The Signer — and thus the signatory — is accountable and cannot technically repudiate being the origin of the document in its current state as the requirements assume that 3^{rd}-party-verifiability is given. In the case of non-public interactive forms of accountability of the MSS (INT) the Verifier needs to carry out an interaction with a party with access to the signatory's secret signature generation key. This deviates from the accountability provided by legally accepted CDSS, which offer non-interactive public accountability (PUB). An extensive discussion on legal problems inherent to not achieving PUB is given in Subsections 6.4.2, 6.4.4 and 6.4.5. For this reason, this analysis follows the recommendation[960] in Requirement 4.1[961]. Given an MSS that offers ACA – ≥1CD – PUB and that it was used to identify the absence of subsequent modifications, Case 1 technically resembles the integrity and authenticity protection offered by a signature that is valid under the Signer's public key created by a legally recognised CDSS. The Verifier can clearly use the positive validation result as a proof that the signed document is still the original as originated from the signatory. As a result, Case 1 is said to provide the same probative value as a valid Signer's signature created by a CDSS:

An unmodified signed electronic document signed by an MSS with ACA – ≥1CD – PUB (Case 1) makes it become prima facie evidence or evidence with a legal presumption of authenticity.
The result is captured in full detail as Evaluation Result 3 (see Sec. 17.6).

17.5.2.3. Case 2: Valid signature and detection of occurred subsequent modifications yields document has been modified, but no Signer consent for authorization was given.

In Case 2 the Sanitizer carried out allowed subsequent modifications; the Verifier clearly establishes this fact due to ≥1CD integrity. However, in Case 2 the signatory's consent for this authorization cannot be proven by the Verifier to a third party, thus this usage of an MSS does not fulfil Requirement 2[962], which postulates that the mechanism MUST produce a verifiable proof of the Signer's consent to scope and

[959] These additional mechanisms are based on, but by no means limited to, the mechanism's technical properties. See for example the discussion of the accountability requirements in Sec. 7.4.4 on page 176.

[960] Indicated by using SHOULD not MUST in the formulation of the requirement.

[961] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1 on page 173).

[962] Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier) (see Sec. 7.2).

strength of the integrity and authenticity protection. Thus, signing the document did not create a legal blanket statement. As discussed in Sec. 4.5.5 on page 98 this would have allowed to hold the signatory responsible for the occurred subsequent modifications on the basis that the signatory authorized them with consent. If this is not the case the the signed document's probative value is not increased.

Note, Case 2 was introduced to explicitly convey that the consent — and implicitly also the ability to control the integrity protection — of the signatory is legally important (see Requirement 2[963] regarding consent and Requirement 1[964] regarding control). This thesis wanted to stress that a failure of this would immediately result in a loss of an increased probative value.

Case 2 is out of the technical scope of an MSS with ACA– ≥1CD –PUB integrity protection mechanisms. This case is however reached in at least the following cases:

2.1 The sole integrity protection mechanisms itself, i.e. the MSS, worked flawlessly, but technically flanking mechanisms failed. For example assume that the user interface did not convey to the signatory that with the signature created subsequent changes would be authorized. This is out of the scope and falls into solving the presentation problem and building systems and processes that allow to establish a technical proof of intentional signing (see Sec. 1.4.3 on page 17).

2.2 The integrity protection mechanism has flaws not known to the signatory. Thus, the mechanism would fail to achieve Requirement 1[964] and thus allow adversaries to carry out modifications that the signatory deemed as unauthorized, but which the scheme did by classify as authorized. If this flaw is known to the Verifier at the time and if the signatory could explain that (1) it could not know this and (2) the document in question could be created through the attack then this can be used as a doubt against evidence. Again, this is out of the technical scope that assumes that the algorithms are not cryptographically flawed and fulfil the mandatory Requirement 7.1[a] and thus achieve cryptographic protection against a forgery. A comparable doubt against evidence is discussed in Sec. 17.7.1.

[a] Requirement 7.1 (Mechanism MUST achieve cryptographic protection against a forgery) (Sec. 7.7.1).

17.5.2.4. Case 3: Valid signature and detection of occurred subsequent modifications yields document has been modified and Sanitizer is non-interactive public (PUB) accountable

In Case 3 the Sanitizer carried out allowed subsequent modifications. The occurrence of such an authorized subsequent modification is differentiated because the detection capability of the MSS analysed here is ≥1CD. Additionally to the mandatory Requirements 1–7 and the recommended Requirement 3.3 and Requirement 4.1 the MSS must fulfil Requirement 4.2[965] which states that a mechanism may achieve non-interactive public accountability of the sanitizer. Case 3 is reached when the Sanitizer did indeed carry out an authorized subsequent modification, including touching the message[966]. Further, Case 3 inherits the need from Case 1 that due to non-interactive public (PUB) Sanitizer- and Signer-accountability the Sanitizer becomes technically accountable. Henceforth, the evaluation thus assumes an MSS that achieves ACA – 1CD – PUB integrity. Analog to the argumentation for an increased probative value for Case 1, an MSS signature that verifies after a Sanitizer's subsequent modification and thus corroborates that this modification has occurred gives the Verifier a 3^{rd}-party verifiable proof of the Sanitizer being the origin because the usage of $\mathsf{sk}_{\mathrm{san}}$ can be established publicly and thus the Sanitizer is accountable.

Sanitizer-accountability is as strong as Signer-accountability following the definition given in Sec. 11.6.6. It existed for SSS, but the property did not exist in previous literature for RSS and got introduced in this thesis as ARSS (see Sec. 14.1).

[963] Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier) (see Sec. 7.2).

[964] Requirement 1 (Mechanism MUST protect content, structural and referential integrity as requested by the Signer) (see Sec. 7.1).

[965] Requirement 4.2 (Mechanism MAY achieve non-interactive public accountability of the Sanitizer) (see Sec. 7.4.2 on page 175).

[966] See Sec. 3.4.3.

A document with a valid signature generated by an MSS with ACA – 1CD – PUB allowing to identify the case 3 makes technically the same statement regarding authenticity and integrity of the signed document as a valid signature by the Sanitizer generated by CDSS with NCA–1CD–PUB integrity. This is achieved by the MSS being able to corroborate a proof to third parties that at least one authorized subsequent change by a Sanitizer endorsed by the Signer got identified.

With respect to the research question the probative value offered is classified as *increased* for the following two reasons: First, the probative value is increased because the Verifier can corroborate a 3^{rd}-party verifiable proof that the signatory consented to and authorized the subsequent modification. Hence, following the ideas of a legal blanket statement (see Sec. 4.5.5 for the legal background and Sec. 18.7 for an example application), the original signatory — equal to the Signer — will be legally liable for the modified document. Second, the Sanitizer is technically accountable for the signed document as if it was directly signed by an CDSS. The party in the role of the Sanitizer can thus be legally approached as being the only signatory comparable to case 1 of the CDSS. This allows assigning liability to the party who did the subsequent edits — the Sanitizer. As discussed the public form of accountability increases legal certainty (see Sec. 4.6.4) in general and public Sanitizer-accountability allows for publicly attributing the Sanitizer[967]. Therefore, this thesis sees additional value in using MSS (like $pubacc\mathcal{SSS}$ and $pubacc\mathcal{RSS}$) which allow holding the Sanitizer accountable towards a third party.

MSS with ACA – ≥1CD – PUB can provide up to evidence with a legal presumption of authenticity for a Sanitizer modified signed electronic document but appoint the Sanitizer as the signatory.
The technical protection offered by the MSS

- is equal to that of a CDSS generated by the signatory (prima facie evidence or evidence with a legal presumption of authenticity), and
- it establishes a legal blanket statement (see Sec. 4.5.5) by the signatory

if

- the MSS achieves the integrity functionality and mandatory Requirements 1–7 and additionally
- the MSS achieves the recommended Requirement 3.3[a], and as well
- the MSS achieves the recommended Requirement 4.1[b], and
- the MSS achieves the optional Requirement 4.2[c] which demands Sanitizer-accountability, and
- the integrity validation result is that the signature is valid under the Signer's public signature verification key, and
- only authorized subsequent modifications were detected due to offering ACA – ≥1CD – PUB integrity [d], and
- the MSS is registered to fulfil Requirement 7.3.

See the doubt that the algorithm is not accredited given in Sec. 17.7.2 for a further discussion of the implications of non-existing legal accreditation of an algorithm.
See Evaluation Result 5 in Sec. 17.10 for an evaluation under which circumstance the signature provides prima facie evidence or evidence with a legal presumption of authenticity.

[a] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) (see Sec. 7.3.3).

[b] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1).

[c] See Sec. 7.4.2 on page 175.

[d] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms) (see Sec. 7.7.3).

17.5.2.5. Case 4:

If the MSS used did not offer Sanitizer-accountability as a public and 3^{rd}-party-verifiable property, then a mechanism would not be able to offer signature verification outcomes that will classify as Case 0, 1, or 3. However, the mechanism might still fulfil Signer-accountable to the same standard as in Case 3, but just not offer public non-interactive Sanitizer-accountability; those mechanisms' outcomes are classified into Case 4.

[967] Namely, this public form fullfills the "Verfikationsfunktion" [147] and allows to identify the Sanitizer, as in "Bei der Technik der qualifizierten elektronischen Signatur kann der Adressat in einfacher Weise überprüfen, ob die Signatur mit dem privaten Signaturschlüssel des Schlüssel-Inhabers erstellt worden ist und ob der signierte Text nachträglich verändert worden ist." [147]

Due to the missing Sanitizer-accountability the Verifier seeing a Case 4 result would **not** be able to corroborate a 3^{rd}-party-verifiable proof that a Sanitizer is accountable for the document that was subsequently modified in only authorized ways. This inability of offering non-interactive public accountability for the Sanitizer prohibits the Verifier from establishing any more than that the document is a valid embodiment of a blanket statement. This case typically happens with RSS where the sanitization is a public operation. Then, even if an RSS offers to detect with public 3^{rd}-party-verifiability that a subsequent modification has happened and that it was authorized, the Verifier will not be able to hold a single legal entity accountable for having done the redaction. Still, a valid signature verification outcome of an RSS with ACA – ≥1CD – PUB will still corroborate that the Signer has endorsed the document to be sanitized. If the public detection of subsequent modifications identified that no subsequent modification has occurred, then the outcome is of Case 1. If the public detection of subsequent modifications identified that an authorized, but no unauthorized, subsequent modification has occurred, then the outcome is of Case 4.

However, as due to missing Sanitizer accountability this signature's transferability[968] is limited, this thesis sees it as not comparable with the proof generated for a Verifier when verifying a signature generated by a CDSS. Especially, Case 4 is **not** equivalent to a CDSS of the Sanitizer, as is Case 3.

MSS with ACA – ≥1CD – PUB can provide an increased probative value as it identifies the document as a valid embodiment of a blanket statement issued by the signatory (see Evaluation Result 4 in Sec. 17.9).

The technical protection offered by the MSS establishes that the signed document is based on a legal blanket statement issued by the signatory if

- the MSS achieves the integrity functionality and mandatory Requirements 1–7 and additionally
- the MSS achieves the recommended Requirement 3.3[a], and as well
- the MSS achieves the recommended Requirement 4.1[b], and
- the MSS fails to achieve the optional Requirement 4.2[c], i.e. it does not achieve Sanitizer-accountability but achieves Signer-accountability, and
- the integrity validation result is that the signature is valid under the Signer's public signature verification key, and
- only authorized subsequent modifications were detected due to offering ACA – ≥1CD – PUB integrity.

[a] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) (see Sec. 7.3.3).

[b] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1).

[c] See Sec. 7.4.2 on page 175.

17.5.3. Additional cases due to block-level properties

The above five cases (Sec. 17.5.2) are for mechanisms that offer ACA – ≥1CD integrity on a message-level. They represent a basic set, which also covers the cases identified as interesting in the light of the legal probative value introduced by MSS — the scope of this thesis.

If the scheme supports ACA – ≥1CD integrity with block-level detection of subsequent modifications, then additional cases can be differentiated. For example, Case 3 and Case 4 could be split, and have an additional sub-case due to a positive answer to the question if the Verifier would be able to further distinguish which block of the document has been modified. Henceforth they are denoted as Case 3.b and Case 4.b. A verification result in Case 3.b would enable the Verifier to identify clearly those block(s) which have verifiably not been subsequently modified and those that have been modified with authorization by an accountable Sanitizer. The former block(s) can be deemed original and thus can be handled following Case 1 from Sec. 17.5.2. The latter block(s) are clearly identified as non-original and are handled like the message was in Case 3 from Sec. 17.5.2. Analog for Case 4.b: Case 4.b would enable the Verifier to identify clearly those block(s) which have verifiably not been subsequently modified and those that have been modified with authorization. The former block(s) can be deemed original and thus can be handled following Case 1 from Sec. 17.5.2. The latter block(s) are clearly identified as non-original and are handled like the message was in Case 4 from Sec. 17.5.2.

[968] Note, transferability is used in the rather intuitive meaning of *Katz and Lindell* (see Sec. 5.3.7 on page 128).

17.6. Evaluation Result 3: A signature by an MSS with ACA – ≥1CD – PUB, that yields that the signed document is unmodified, is a qualified electronic signature following Regulation 910/2014

On top of Evaluation Result 2 which stated that MSS are ‘elementary’ electronic signatures following Regulation 910/2014, this section will show that an MSS with ACA – ≥1CD – PUB integrity generates a signature that meets the legal requirements of a *qualified electronic signature* according to Regulation 910/2014 if the verification result is ranked as Case 1. The following Tab. 32 lists the requirements that are defined — mainly in Article 26 and Article 3 section 11 and 12 Regulation 910/2014[969] — and a brief reason why the thesis evaluates that the MSS fulfils this requirement and thus can generate a qualified electronic signature following Regulation 910/2014.

***Interpreted*[969] Requirements from Art. 26 and Art. 3 (11)–(12) Regulation 910/2014**	**MSS**	**Note**
(1) *created signature* is uniquely linked to the signatory;	✓	verification with pk_{sig} yields that sk_{sig} was involved in creation of σ by fulfilment of Requirement 4[970]; pk_{sig} is linked by to the signatory, e.g. via certificates and PKI, by fulfilment of Requirement 5[971];
(2) *created signature* is capable of identifying the signatory;	✓	
(3) *signature* is created using electronic signature creation data *(sk_{sig} or sk_{sig})* that the signatory can, with a high level of confidence, use under his sole control; and	✓	asymmetric cryptography enabled to guard sk_{sig}, e.g. in a HSM see also (6), while the public key pk_{sig} can be shared and certified
(4) *signature* is linked to the data signed therewith in such a way that any subsequent change in the data is detectable.	✓	achieved through ACA/NCA – ≥1CD –PUB in fulfilment of Requirement 3[972];
(5) *signature* is created by a qualified electronic signature creation device *(QSCD)*	✓	executing functions using the secret key inside a secure smart card in fulfilment of Requirement 5[971]
(6) *signature* is based on a qualified certificate for electronic signatures.	✓	pk_{sig} corresponding to sk_{sig} has been linked to the signatory using a qualified public key certificate (Sec. 2.11) issued by an entity that had followed a legally required process

Table 32. Schemes achieving to have a verification outcome in Case 1 are qualified electronic signatures following Regulation 910/2014; *italics* it used to identify interpreted wording, original in Footnote 969.

The list of mandatory requirements given previously are fulfilled by schemes like $pubacc\mathcal{SSS}$, $pubacc\mathcal{SSS}^{Block}$ and $pubacc\mathcal{RSS}$ as mentioned in Sec. 17.3.

An unmodified signed document that was created by an MSS that fulfils the mandatory Requirements 1–7 and offers non-interactive public accountability (PUB) of the Signer (Case 1) is a qualified electronic signature according to Article 3 Section 12 Regulation 910/2014and makes the document become prima facie evidence or evidence with a legal presumption of authenticity.

The technical protection offered by the MSS is equal to that of a CDSS (prima facie evidence or evidence with a legal presumption of authenticity)[a], if

[969] Original texts: “‘advanced electronic signature’ means an electronic signature which meets the requirements set out in Article 26;” [Art. 3 (11) Regulation 910/2014].
“‘qualified electronic signature’ means an advanced electronic signature that is created by a qualified electronic signature creation device, and which is based on a qualified certificate for electronic signatures;” [Art. 3 (11) Regulation 910/2014].
“An advanced electronic signature shall meet the following requirements:
(a) it is uniquely linked to the signatory;
(b) it is capable of identifying the signatory;
(c) it is created using electronic signature creation data that the signatory can, with a high level of confidence, use under his sole control; and
(d) it is linked to the data signed therewith in such a way that any subsequent change in the data is detectable.” [Art. 26 Regulation 910/2014].

[970] Fulfilment of Requirement 4: Based on detection of any occurred subsequent modification (≥1CD) achieve Signer accountability (Sec. 17.3.4).

[971] Fulfilment of Requirement 5: Linkage towards the legal signatory (Sec. 17.3.5).

[972] Fulfilment of Requirement 3: Any subsequent modification is detected (≥1CD), either INT or PUB (Sec. 17.3.3).

- the MSS achieves the mandatory prerequisites from Requirements 1–7, and additionally
- the MSS achieves the recommended Requirement 3.3[b], and as well
- the MSS achieves the recommended Requirement 4.1[c] and
- the integrity verification result is that the signature is valid under the Signer's public signature verification key, and
- no subsequent modifications were detected due to offering ACA – ≥1CD – PUB integrity, and
- the MSS is registered to fulfil Requirement 7.3[d] .

See the doubt that the algorithm is not accredited given in Sec. 17.7.2 for a further discussion of the implications of non-existing legal accreditation of an algorithm.
See Evaluation Result 5 in Sec. 17.10 in order to identify the circumstances when a signed document is legally treated as evidence with a legal presumption of authenticity and when it is treated as prima facie evidence.

[a] That is unless the law explicitly demands a handwritten document.

[b] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) (see Sec. 7.3.3).

[c] Requirement 4.1 (Mechanism MUST achieve detection of any occurred authorized and unauthorized modification including achieving accountability of the Signer and including achieving technical non-repudiation of signature creation; Mechanism SHOULD achieve non-interactive public accountability of the Signer) (see Sec. 7.4.1).

[d] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms) (see Sec. 7.7.3).

17.7. Discussion of possible doubts against evidence created by MSS

This section discusses possible doubts against evidence ('Beweiseinreden') that could be cast on a document signed with a signature created by an MSS when it is used as evidence in legal proceedings. The doubts aim at reducing the probative value generated by the electronic signature[973]and are based on an analysis given by *Grigorjew* [219] for the probative value of non-qualified electronic signatures in the former German signature law following Directive 1999/93/EC. The most prominent doubts from [219] have been selected and transcribed to attack MSS under Regulation 910/2014.

Each one of the following doubts calls the evidence of the signed document in question. If the party using the signed document's increased probative value in their favour can not dispel those doubts, the doubts can defeat the increased probative value induced by the MSS as explained in Sec. 17.3. Hence, the goal is to refute such doubts, as the probative value induced by an digital signature that is not achieving the requirements needed to be called a qualified electronic signature will result in an individual variance of the probative value according to the analysis of *Grigorjew* [219][974].

It is important to discuss possible doubts and objects: The previous evaluation on the probative value (see Chapter 4) assumed that the actual fact that the document bears an electronic signature with the legally required level of strength is not disputed in front of the court. Only if there is no dispute about the fact that a qualified electronic signature in accordance with the Regulation 910/2014 has been generated and that it has been positively verified, the signature gives the integrity and origin authentication necessary for the rules of evidence to apply. Especially, only then statutory evidentiary presumption can be awarded as stated in the legal texts[975].

[973] *Grigorjew* [219] called them "Beweiseinreden" [219], which this thesis did transcribe with 'doubts against evidence' or 'to question evidence'. A translation to 'demurrer of evidence' is not correct as this term aims to describe actions that try to forbid the evidence being admitted, following *Dietl and Lorenz* this translates to "Einrede des unzureichenden Beweises" [152]. Another translation of "Beweiseinreden" [219] that again steers into that direction is "objections to the admission of evidence" that can be found in `https://www.dict.cc/?s=Beweiseinrede` [last accessed Jun. 2017]. However, as described this is not the notation that was intended in this context.

[974] In German the analysis states "Die Analyse der Signaturverfahren unter dem Niveau einer qualifizierten elektronischen Signatur hat ergeben, dass der Gestaltungsspielraum und somit das Sicherheitsniveau sehr unterschiedlich sind. Dieses reicht von niedrig bis relativ hoch. Eine pauschale Aussage über die Sicherheit von Signaturverfahren unterhalb des Niveaus qualifizierter elektronischer Signaturen konnte demzufolge nicht getroffen werden. Jedes einzelne Signaturverfahren muss für sich allein betrachtet und bewertet werden." [219].

[975] This was discussed in Germany for the former electronic signature act (SigG), e.g., by *Fischer-Dieskau et al.* [195] and *Fischer-Dieskau et al.* [193]. In German the analysis stated: "Das Vorliegen einer akkreditierten oder qualifizierten elektronischen Signatur begründet somit aus sich heraus nicht automatisch das Vorliegen der Voraussetzungen des Anscheinsbeweises. Vielmehr müssen sich diese erst aus der Prüfung der Signatur nach dem SigG ergeben. Ist ihr Vorliegen vor Gericht zwischen den Parteien unstreitig, hat das Gericht dies als Tatsache in seine Beweiswürdigung und Entscheidung einzubeziehen. Ist die Signaturstufe jedoch strittig, so muss der Beweisführer ihr Vorliegen zur Überzeugung des Gerichts nachweisen." [195].

To jump to the conclusion, the doubts are either not specific to the use of MSS and thus are a problem to CDSS as well as MSS; or they challenge assumptions made in this thesis due to the scope being the technical cryptographic mechanisms. For the latter case this section explicitly points out which assumption from the thesis is challenged by the question to the evidence.

17.7.1. Doubt: Algorithm not acceptably strong for a qualified electronic signature

An doubt can be made against the algorithms offer a suitable level of cryptographic security[976] [219]. As raised in this thesis as Analysis Result 3.3[977] the Regulation 910/2014 appoints that technical specifications are to be set out that identify suitable algorithms[978]. While legal texts directly or indirectly point to standards and catalogues, i.e., [85].[979], they are in generally formulating the requirements in technology-neutral language. Hence, an expert opinion would be required to attest that the cryptographic strength that the algorithms employed in the MSS are equally strong to those listed in the catalogues. This thesis draws this connection in the evaluation of the $pubacc\mathcal{SSS}$ and $pubacc\mathcal{RSS}$ regarding their strength in Sec. 17.3.7. An algorithm would be not legally strong if an adversary could create a forgery with 'relatively justified effort'[980][409]. In the case of MSS this thesis showed provably secure constructions and assumes that an expert opinion would come to the same conclusions.

An argumentation to dispel the doubt is as follows: The cryptographic building blocks from which the MSS is constructed already are legally accepted as strong enough. In this thesis many of the constructions can be instantiated from signature schemes that are legally accredited, e.g., $pubacc\mathcal{SSS}$ they can be based upon an UNF-CMA signature scheme without needing additional hash-functions. As such the doubt would result in an expert opinion which could follow the arguments regarding the strength of the algorithms in Sec. 17.3.7. Note, as explained by *Grigorjew* [219] the bar will judge this at the court's discretion and conviction[981] and thus will, additionally to the technical expert opinion, base the final decision on the motives and the availability of the needed cryptographic and technical skills of the accused [219, p. 215–216].

17.7.2. Doubt: Algorithm is not accredited

An doubt can be made because currently none of the algorithms for RSS and SSS are listed in the legally required documents. Indeed, MSSs miss the optional Requirement 7.3[982]. However, this might change in future as RSS are just starting their route to international standardisation[983]. While MSS clearly fail to fulfil Requirement 7.3 it is important to note that the used mechanisms are legally required to achieve a certain functionality described in technology-neutral language (see Analysis Result 3.3[984]).

If these were listed in the most-recent catalogues, e.g. in the German 'Algorithmenkatalog' (see Sec. 5.3.1) the algorithms are already attested a legally sufficient strength and an expert opinion would not be necessary to attest the increased probative value [219, p. 212–213]. However, if not listed, the question is as follows: Do they comply with requirements set forth in the Regulation 910/2014 and respective additional member state legislation[985]. Those legal texts directly or indirectly point to standards

[976] In German this is also termed "sicherheitsgeeignet" [219, p. 213]. *Grigorjew* states these objections under "6.1.3.1.5.1 Verwendung sicherer Hash- und Signaturalgorithmen" [219, p. 212].

[977] Analysis Result 3.3: Electronic signature legislation texts describe required functionality in technology-neutral language (see Sec. 4.6.3.3 on page 104).

[978] This is in line with the work of *Grigorjew* from which this doubt is transposed, which was originally pointing to German signature law written before Regulation 910/2014, namely "§ 17 Abs. 1 SigGund § 15 Abs. 1 SigVOi. V. m. Anlage 1 Abschnitt I Nr. 2 SigVO" [219, p. 213].

[979] *Grigorjew* [219] related to this fact in footnote 859 [219, p. 260].

[980] Translated from the German "verhältnismäßigem Aufwand" [409, p. 166].

[981] In German "Freie Beweiswürdigung" [99].

[982] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms) (see Sec. 7.7.3).

[983] ISO New Work Item Proposal (NWIP) for 'Redaction of Authentic Data' was submitted for national voting ballot in late 2017, voting will close in March 2018.

[984] Analysis Result 3.3: Electronic signature legislation texts describe required functionality in technology-neutral language.

[985] *Grigorjew* [219] related to the former German SigG implementing the predecessor of Regulation 910/2014 by referencing "§ 17 Abs. 1 SigG und § 15 Abs. 1 SigVO i. V. m. Anlage 1 Abschnitt I Nr. 2 SigVO" [219, p. 213].

and catalogue, i.e., [85].[986] Here, an expert opinion would be required to attest that the cryptographic strength that the algorithms employed in the MSS are equally strong to those listed in the catalogues. The basis of that was provided in the evaluation of MSS regarding its cryptographic strength in Sec. 17.3.7 As noted in the argument, the constructions can be instantiated from signature schemes that are legally accredited, and such the assumption that this argument holds is being made in the argumentation for an increased probative value in Sec. 17.3. Namely, the only difference remaining is that the optional Requirement 7.3[987] is not fulfilled.

17.7.3. Doubt: Keys not managed securely

Grigorjew notes that a legal argument for a heightened probative value requires that the keys are securely managed on a qualified signature-creation device (QSCD)/secure signature-creation device (SSCD) [219, p. 231] and that the key(s) can be correctly linked to the legal signatory. Of course this can be called into question to cast a doubt on the evidence. Generally this can be called into question for every advanced, i.e. not-qualified, electronic signature, because it calls especially the organisational and technical management of the link between the legal signatory and the technical key-holder into question. For all categories but qualified electronic signatures there are not technical processes legally pre-scribed in Directive 1999/93/EC [219] — same for Regulation 910/2014.

An argumentation to dispel the doubt is as follows: The question suggests problems in the accompanying additional organisational processes resulting in the scheme not fulfilling the accountability, which is legally crucial as pointed out in Analysis Result 12[988] and Analysis Result 5[989]. However, this thesis showed that MSS can indeed be constructed such that the overarching requirements for strong identification and authentication and control of use can be fulfilled. Requirement 5[990] captures the technical basis for this as it requires that the scheme has public signature verification keys. MSS achieve this (see Sec. 17.3.5) and thus an overarching public key infrastructure can tie the public verification key to the legal signatory. MSS further allow the keys to be maintained inside secure smart cards (see Sec. 17.3.6) and thus fulfil Requirement 6[991] and Requirement 5[990] which shall dispel this doubt — and further related ones raised by *Grigorjew*[992]. As many of the proposed constructions can be instantiated from signature schemes that are legally accredited and because the thesis has proven for schemes like $pubaccSSS$ that they can be run on a qualified signature-creation device (QSCD)[993]. With the help of a secure smart card (acting as QSCD) and a working PKI the public keys of an MSS can be managed in the same way as for legally accepted signature mechanisms.

17.7.4. Doubt: Keys not secret

In the following it is assumed that the above objections of a not working public key infrastructure cannot be successfully made, i.e., that the public key certificate in question is actually qualified and non disputed. Still, yet another vector to discredit the increased probative value is to object that the secret signature key has been compromised (in German "6.1.3.3.3 Kompromittierung des Signaturschlüssels" [219, p. 234]). When Security Assumption 1 (Secret keys are not known to the adversary) still holds, a com-

[986] *Grigorjew* [219] related to this fact in footnote 859 [219, p. 260].

[987] Requirement 7.3 (Mechanism MAY be already registered in legally accepted sources for acceptably strong cryptographic algorithms).

[988] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

[989] Analysis Result 5: Directive 1999/93/EC and Regulation 910/2014 technically require origin authentication that includes accountability and non-repudiation of the signatory (see Sec. 4.6.5 on page 105).

[990] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate) (see Sec. 7.5).

[991] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

[992] *Grigorjew* lists additional doubts ('Beweiseinreden' in German): (1) incorrect name in the certificate (in German "6.1.3.3.1 Nennung einer falschen Person im Zertifikat" [219, p. 232]) and (2) lost signature generation key (in German "6.1.3.3.2 Verlust des Signaturschlüssels" [219, p. 234]) However, those doubts are only against the actual linking of the legal signatory to the public signature verification key and can be refuted based on the fulfilment of Requirement 6[991] and Requirement 5[990] coupled with the two overarching assumptions regarding key management: Security Assumption 1 (Secret keys are not known to the adversary) and Security Assumption 2 (All entities can retrieve trustworthy public keys unique for each alleged entity when needed).

[993] See Sec. 13.10 for SSS and Sec. 14.15 for RSS.

promise would require that either (1) the mechanism is not safe against the extraction of the secret signature generation key from the the public verification key or the signature or (2) the secret can be extracted from the secure signature-creation device (SSCD). The MSS schemes proposed in this thesis have been proven to be secure and especially offer unforgeability that would be possible if the secret key would leak. this means that the qualified signature-creation device (QSCD)/secure signature-creation device (SSCD) is not working, i.e., that the secret signature generation key is no longer unique[994]. As the MSS mechanisms that are proposed are instantiated upon secure accepted signature schemes they can operate using the exact same environment for key management as legally accepted schemes. Hence, the argument was made that MSS can fulfil Requirement 6[995] and Requirement 5[996] and thus induce an increased probative value for the document they signed. Here, state of the art hardware security modules (see Definition 29) allow to defend the challenging party and uphold an increased probative value [219, p. 236].

17.8. Discussion of impact on general limitations of this evaluation

This section will briefly highlight two more general limitations, which are due to this evaluation being general and not an expert opinion for a concrete technical system and legal limits to a delegation.

17.8.1. General limitation: The thesis analysed the low levels of cryptographic properties of RSS and SSS; its not a full expert opinion

Grigorjew proclaimed that in cases where not all requirements for a qualified electronic signature are clearly given, the needed expert opinion is very important. [219, p. 238][997] In this respect, the thesis gave a detailed integrity notion that allows to exactly compare the offered properties. This allows comparing the offerings of different MSS among each other and allows for a comparison to the legally widely accepted CDSS. The mathematical proofs for the important questions regarding integrity protection and Signer-accountability have been provided for the proposed constructions.

On top of that, an expert opinion for a concrete technical implementation, even if it is based on one of the constructions, will need to describe that the technical realisation and the concrete parameters chosen for the algorithms offer the sufficient level of cryptographic security, especially towards the properties immutability, unforgeability and accountability. Hence, the author of this thesis is well aware that this is not a full expert opinion[998].

17.8.2. General limitation: Direct representation even by a secure malleable signature schemes is not always allowed

The legal system in general foresees and allows delegation of one party's privileges in juristic acts. An example is to delegate the signing of a contractual agreement, to a third party, called an agent. The law further distinguishes between direct and indirect representation. "Direct representation exists where an agent acts 'in the name of' a principal" [102] and it is obvious for all parties, including the third party, that the agent acts on behalf of the principal. In cases of direct representation the agent, if it had the legal authority and did act correctly, plays no role in the resulting legal relation between the principal and the third party. Indirect representation can be divided into two forms: The first form can be called " 'undisclosed agency' " [102], "[t]he second category of indirect representation can for convenience be termed 'commission agency'." [102] In the first form the third party is not aware of the agent having the role of the principal's agent. The agent acts in his own name towards the third party and thus "[...] the

[994] Here, 'not unique' means that is has been replicated.

[995] Requirement 6 (Secret key-dependent algorithms of the mechanism MUST execute in a secure-signature-creation device (SSCD) or a qualified electronic signature creation device (QSCD)) (see Sec. 7.6).

[996] Requirement 5 (Mechanism MUST expose a public verification key, which can be linked to a natural (or legal) person by a qualified certificate) (see Sec. 7.5).

[997] In German "[...] im Streitfall [kann] der Sachverständigentätigkeit eine enorme Bedeuteutung zukommen [...]" [219, p. 238].

[998] In absence of any non-prototypical installation to use as an evaluation target it was also never intended to be a full expert opinion.

third party neither knows nor has reason to know that the intermediary acts as an agent." [102] However, there is a second contract between the agent and the principal. In the latter form the "[...] intermediary is to act 'in his own name': [H]e is to contract with third parties in a personal capacity" [102] and thus "[n]o contractual relationship is created between third party and principal." [102].

The detailed legal effects, especially in contract law, of the different forms of representation are not of further interest, but they have been introduced to show that the concept of MSS falls into direct representation, e.g., "direkte Stellvertretung" in § 164 BGB. It is the Signer's signature verification key that can be used to verify the signatories involvement. Hence, the delegation inherent to the MSS does not hide the principal and in the German BGB § 164 (1) thus applies stating that "A declaration of intent which a person makes within the scope of his own power of agency in the name of a principal takes effect directly in favour of and against the principal. It is irrelevant whether the declaration is made explicitly in the name of the principal, or whether it may be gathered from the circumstances that it is to be made in his name. [...]" [§ 164 BGB].

In Sec. 4.5.5 this thesis already stated that an MSS technically can give the Sanitizer the legal authority, because the policy of admissible authorized subsequent modifications can be seen as a codified "letter of authorization" given by the Signer following § 172 BGB. Then, the Sanitizer has the role of an agent and the Signer is the principal. Due to the cryptographic properties of unforgeability and immutability this policy can not be changed. Whenever the Sanitizer modified block(s) of a malleably signed document with authorization, it does so as a direct representative of the Signer. As discussed earlier in Sec. 4.5.5.1, due to direct representation, all acts done by the agent (on behalf of the principal who authorized the agent) "[...] effect directly in favour of and against the principal [...]" [§ 164 I 1 BGB] who authorized the agent. Thus, the Signer using a malleable signature scheme, becomes liable as legal signatory for any validly signed statements generated by authorized modifications done by Sanitizer (s).

Note, however that representation might be explicitly excluded by a binding contract or directly by the law[999]. In the latter, this can be found in the laws' texts or it might be excluded due to the nature of the juristic act[1000]. Hence, just because the use of a malleable signature scheme makes a delegation technically possible, it might not be legally allowed. However, especially with respect to the legal risks of engaging with an agent rather than with the principal this thesis would like to point out the following: The third party is in the technical role of the Verifier and will be able to discover that, due to the special algorithms involved in MSS, there is a potential for authorized modifications[1001]. Thus, the third party will be able to detect that a valid signature of a principle generated by an MSS introduces an agent, but it will remain a direct representation.

17.9. Evaluation Result 4: A signature with an MSS with ACA – ≥1CD –PUB integrity but no Sanitizer-accountability generates a legal blanket statement

The German laws have the legal concept of blanket statements. In a nutshell, this describes that a statement which is underspecified with the consent of the issuing entity, becomes legally binding in its fully specified version against the entity that issued the underspecified statement.

The legal blanket statement was described in detail in Sec. 4.5.5. It led to the formulation of Analysis Result 7[1002] that in order to gain legal accountability of the signatory even in the case of a subsequent authorized modification, the proof of the signatory's consent to that authorized modification must be reproducible. This has been formulated as Requirement 2[1003] and the fulfilment of this by all the MSS

[999] In German: "Gesetzliche Vertretungsverbote" found for example in § 1641 BGB.

[1000] Examples for excluded representation in German law are "The testator may make a will only in person." [§ 2064 BGB], or parents cannot use the power of agency which they have for their minors to have them married (translated from the German "verheiraten").

[1001] Even if an MSS does not allow to detect future potential changes (NFD), as discussed in Sec. 6.3.2, than one can not identify their absence and must legally assume their presence from the fact that an MSS was used, which NFD still allows to establish.

[1002] Analysis Result 7: Legal accountability of the signatory in case of an authorized subsequent modification requires a verifiable consent to a delegation of signing rights (see Sec. 4.6.7 on page 107).

[1003] Requirement 2 (Mechanism MUST produce verifiable proof of the Signer's consent to scope and strength of the integrity and authenticity protection; this proof SHOULD be non-interactively and publicly verifiable by the Verifier) (see Sec. 7.2).

has been discussed in Sec. 17.3.2. Due to the assumption that the chosen mechanism and its configuration together are the integrity policy. As this technically defines if and what subsequent modifications are detectable and which subsequent modification results in documents that bears a valid signature of the Signer. Further, it is assumed that the limitations of mechanisms are consented to by the signatory due to the concrete mechanism being chosen with the knowledge of its limitations to generate the signature. The configuration used during generation of the signature (technically this includes the ADM) is also covered by the Signer's signature and thus cannot be changed in ways not authorized by the Signer because the signature schemes are immutable and unforgeable. See also the more detailed discussion regarding the Analysis Result 8[1004] on the necessity to define the required integrity protection depending on the concrete application's needs.

17.10. Evaluation Result 5: Rules to award evidence with a legal presumption of authenticity carry over to MSS

As described earlier there are statutory rules of evidence (discussed in Sec. 4.1.1 on page 72) which are found in the German legislation in § 286 paragraph 2 ZPO "statutory rules"[1005] [99]. They assign a probative value for the case that an electronic document is bearing a qualified electronic signature of either prima facie evidence or evidence with a legal presumption of authenticity[1006]. In general the assurance level of the electronic signature being qualified must not be disputed when discussing the probative value awarded by the signature[195].

So in the following, the evaluation assumes that the technical assurance given by the MSS signature amounts to it being recognised as a qualified electronic signature (see Evaluation Result 3). Then statutory rules in the German legal text prescribe the document with an increased probative value. The rules are as follows:

Documents of official authority with qualified electronic signatures have the highest probative value of evidence with a legal presumption of authenticity. Statutory rules laid out in the German ZPO award an evidence with a legal presumption of authenticity to a document of official authority that carries a valid qualified electronic signature (see Sec. 4.2.2.1).

Non-public documents executed by private party with qualified electronic signatures gain a high probative value as prima facie evidence. If a non-public document executed by private party carries a valid qualified electronic signature it can get a high probative value under the regime of § 371 ZPO as evidence taken by visual inspection ("Augenscheinsbeweis"). Again the statutory rules of evidence from the ZPO for paper documents apply to electronic non-public documents executed by private party with qualified electronic signatures and thus the documents are assigned prima facie evidence (see Sec. 4.2.2.2).

For further information see Sec. 4.2.2 and if further interested, see *Roßnagel and Fischer-Dieskau* [415] for an in-depth discussion. These rules in the German ZPO make no difference to whether or not the mechanisms would potentially allow subsequent authorizations (see also Sec. 4.5 on page 95).

The statutory rules of evidence in the German ZPO apply to documents signed with qualified electronic signatures and hence the same rules apply to MSS regardless. Alike for qualified electronic signatures created by a CDSS, the German ZPO assigns non-public documents executed by private party prima facie evidence and documents of official authority get treated as evidence with a legal presumption of authenticity when the qualified electronic signature was produced by an MSS that fulfils the requirements necessary to generate qualified electronic signatures (see Sec. 17.6).

[1004] Analysis Result 8: Applications need to define the equivalence classes of modified and unmodified; a policy to describe what constitutes a modification is assumed to capture all data that is relevant in law (see Sec. 5.5.1 on page 130).

[1005] The English translation was taken from the translation provided in [96]. Because this translation is based on an outdated version of the ZPO it was copied only after carefully checking that there is no literal difference in the German text when compared with the latest version [99].

[1006] 'Anscheinsbeweis der Echtheit' oder 'gesetzliche Vermutung der Echtheit' translated to 'evidence with a legal presumption of authenticity' according to the PONS [398] dictionary http://en.pons.eu/translate?q=Gesetzliche+Beweisvermutung&l=deen.

17.11. Evaluation Result 6: MSS can offer detectability of double sanitized documents created by sanitization or redaction from the same origin, but no ordering can be derived

The following evaluation is made based on the MSS proposed in this thesis and the existing schemes known to the author of this thesis. For the use in practical systems it is interesting to actually provide the opposite of unlinkability: linkability. This allows the Verifier to corroborate if two signed documents have a common predecessor and was codified as the optional[1007] Requirement 8[1008]. The scheme *merge-*$\mathcal{RSS}$ (see Sections 14.11 and 14.12) achieves this property and it is explained how to construct an explicit algorithm for linking. This helps in certain scenarios as detailed in Sec. 18.4, where as an example the linkability increases the legal usefulness of the application of an RSS in a supply chain. However, the order in which the redactions or sanitizations have taken place are not recorded, neither in unlinkable nor linkable MSS. For SSS schemes offering ACA – ≥1CD – PUB integrity this obviously only holds true when comparing sanitized documents among each other and not with the original.

EXAMPLE

Example: Assume an MSS that offers linkability and the original message being $m = ''A''||''B''||''C''$ with all blocks being marked as redactable for an RSS or sanitizable in case of an SSS. Then assume that one creates a message m^1 by sanitizing m to $m^1 = ''a''||''B''||''c''$ and a second message m^2 which was created by first sanitizing m to $m^{2pre} = ''a''||''B''||''C''$ and then m^{2pre} becomes sanitized into the $m^2 = ''a''||''B''||''c''$. The Verifier is not able to see from the signatures that m^2 was sanitized twice. Also when given a third message $m^3 = ''a''||''B''||''d''$ that was obtained by sanitizing m^1 the Verifier can not know from the signatures that it was obtained by sanitizing m^1. All the Verifier learns if the SSS offers linkability is that m^1, m^2, and m^3 are having a joint source document. This holds even if the SSS offers ACA – ≥1CD – PUB integrity and thus the Verifier knows that all three messages are sanitizations and not originals.

For RSS schemes that offer detection where the redactions have taken place and how many, i.e., ACA – BCD – PUB, the number of redactions made are of course visible, but still no temporal order of their occurrence leaks.

EXAMPLE

Example: This time, assume an RSS that offers linkability and the original message being $m = ''A''|| ''B''||''C''$ with all blocks being redactable. Then assume that one creates a message m^4 by redacting m to $m^1 = \blacksquare||''B''||C''''$ in the morning and a second message m^5 which was created by first redacting m to $m^{5pre} = ''A''||''B''||\blacksquare$ at lunchtime and then m^{5pre} becomes sanitized into the $m^5 = \blacksquare||''B''||\blacksquare$ in the evening. A Verifier on the next morning is not able to see from the signatures that m^5 was redacted twice and later than m^4. Also when the Verifier is given another message $m^6 = \blacksquare||''B''||\blacksquare$ the next morning, which was obtained by redacting m^4 at lunchtime, the Verifier can not know that it was obtained by redacting m^4. All the insight the Verifier gains from observing the signatures and the sanitized messages generated by an RSS that offers linkability is that m^4, m^5, and m^6 are having a joint source document. This holds even if the RSS offers ACA – BCD – PUB integrity and thus the Verifier knows that all three messages are redacted and where the redaction has taken place.

A secure MSS may offer the cryptographic property of linkability and thus the Verifier can detect if two documents share a common ancestor. A mechanism with the property of linkability fulfils the optional[a] Requirement 8[1008].

[a] Indicated by the word MAY.

A linkable secure MSS does not offer ordering different sanitized documents by their number of sanitizations. The property of linkability, in order to preserve the higher prioritised cryptographic property of standard privacy, only allows identifying if any two sanitized documents with valid signatures have a common ancestor. Especially in terms of linkability, secure MSS that offer the aforementioned property of linkability do not allow the Verifier to identify the order of creation nor any information of previous sanitizations even if the positions or the number of sanitized blocks is known.

[1007] Indicated by the word MAY.

[1008] Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional)).

17.12. Evaluation Result 7: MSS can be facilitated to prove that a different (or original) version of a sanitized document exists, assuming one is in possession of that version's message-signature pair

The following assumes that a party has additional information about sanitized or redacted blocks or their redacted or sanitized structural positions because the party knows a valid signature-message pair that contains this information. Under these circumstances it may be of legal value that the party can convince a third party which holds a validly signed sanitized or redacted document that the known information is from an ancestor of that document. Assuming an RSS, without loss of generality, then the example is as follows:

EXAMPLE

Example: The party called *Bob* knows $m = ''A''||\blacksquare||''C''$ and a valid redacted signature generated by *Charly*. Assume *Alice* holds $m = ''A''||\blacksquare||\blacksquare$ and the corresponding valid signature. It would be interesting if the RSS enabled *Bob* to convince *Alice* that the third blocks of the message was $''C''$, e.g. by sending $m = \blacksquare||\blacksquare||''C''$ with an adapted valid signature to *Alice*.

This is exactly the functionality this thesis formally described with the algorithm Merge for the *merge-*$\mathcal{RSS}$ scheme (presented in Sections 14.11 and 14.12). This gives the possibility to show what the original or an alternative value was in order to settle disputes. As discussed before in Sec. 17.11, it is not possible for the proposed RSS and SSS of this thesis to identify which of two 'alternative' versions is earlier[1009]. However, given a party knows another version or a differently redacted version, a scheme that offers (explicitly or implicitly) mergeability can help to be used for such 'late revelation of unredacted original content'. The applications of this are manyfold. One application is to publicly — or more openly — release redacted or sanitized information with less information. Later, and only if required, it is possible to release the original contents while being able to prove that those contents existed at the earlier release of the sanitized data. This possibility is facilitated to enable the application in the food supply chain, because it allowed later inspections of the full originals by food authorities, as discussed in Sec. 18.4.

A secure MSS offering mergeability or linkability allows to prove the possession of previously original blocks in the case of an RSS or alternative or original blocks in the case of SSS such that the combined document has a valid signature. The goal is that a party is able to generate a proof that it knew the original or an alternate version matching a provided validly signed message. This requires the MSS scheme to offer standard privacy and immutability and unforgeability as defined in this thesis. In case of an RSS, the party successfully doing the proof must have known the original block and a valid signature on it, otherwise it would be an attack on the privacy. In case of an SSS, the party might not be the Sanitizer but must have been in possession of a message containing an alternative content created by an authorized Sanitizer. In both cases the original or alternative content is revealed. A straw man approach is to use linkability — if offered — as discussed in Sec. 17.11: The party provides both messages and both signatures allowing the third party to corroborate the proof by linking them successfully. If the mechanism offers mergeability, the party can re-insert the original or alternative content and adjust the signature such that the third party corroborates the proof by successful verification of the merged signature.

[1009] Standard solution would be to include an external trusted third party to generate a timestamp.

18 — Applications empowered by private RSS and SSS with an increased probative value

Overview of Chapter 18

Chapter 18 presents applications of sanitizable or redactable signature schemes that are private and offer an increased probative value. This chapter presents selected applications of secure SSS and RSS with different levels of maturity:

- **Full system integration:** Sec. 18.1 presents the highest level of maturity. The *pubacc$\mathcal{SSS}$* scheme, a sanitizable signature scheme, was implemented and fully integrated as web services into a business process execution. The integration of signature activities is modelled using a business process modelling language (BPML). Aided by a graphical user interface (GUI), the workflow designers can specify what documents get signed, sanitized and which verification must be achieved in order to further progress in the process. They also select the dedicated entities that can act as Sanitizers. Firstly, this demonstrates how MSS can be presented in a GUI for modelling a supply chain process. Secondly, the MSS steps planned in the GUI were transformed into BPEL and resulted in actual service calls. The functionality was implemented in Java and made available as a web service during the ReSCUeIT project. In the example workflow defined by the research project ReSCUeIT, sanitizable signatures were applied to waybills and laboratory reports in XML. The resulting signatures are embedded in compliance with the XML DSIG specification [26][1010].
- **Proof-of-concept implementation:** Sec. 18.2 presents a proof-of-concept implementation and is the second level of maturity. The *merge$\mathcal{RSS}$* scheme, a redactable signature scheme, was implemented on three different environments (constrained IoT device, smart phone, cloud server). The implementation shows how an RSS brings end-to-end integrity to data streams flowing from mobile phones or sensors to the cloud while keeping data redactable.
- **Concept:** For several application domains the usage of RSS and SSS is described:
  Sec. 18.3 applies RSS to energy consumption data in a smart grid scenario,
  Sec. 18.4 applies RSS to documentation demands of food and feed producers,
  Sec. 18.5 applies RSS to gather authentic answers to requests for personal data being held in a scenario of several companies executing jointly in a service oriented architecture (SOA),
  Sec. 18.6 applies a mergeable RSS to databases in order to allow splitting of sensitive data into several databases while retaining a verifiable signature over the data inside each database,
  and finally Sec. 18.7 describes the use of SSS using the blank bank cheque as an example scenario.
- **Definitional:** Sec. 18.8 formally defines the notion of contingency or intervenability. This is not an application per se, but the dual of the security goal of integrity as it has been defined by *Rost and Pfitzmann* [421] and termed "contingency" by *Bedner and Ackermann* [33]. Sec. 18.8 describes the use of a secure transparent SSS to construct a contingency protection mechanism[1011].

All applications motivate again, why an increased privacy and with it confidentiality protection is required alongside the need to have a strong and legally valid binding between the signed data and the signatory in many application domains. Many of these applications have been published as separate joint publications [198, 226, 375, 379, 382, 383, 387, 390, 391, 397] (see Appendix A).

[1010] Those XML structures are based on the results that got published as joint work with *K. Samelin* and *J. Posegga* [391] (see Appendix A publication n° 6).

[1011] The result was published in the IFIP Security and Trust Managment (STM) conference [379] (see Appendix A publication n° 14).

18.1. Full system integration: Sanitizable waybills and laboratory reports for increased food safety in food supply chains

This following section is composed with material from the reports of the ReSCUeIT project.[1012] The project's goal was to make supply chains more robust. The reason is that supply chains of today are getting more complex and they are handled by complex IT and communication systems. ReSCUeIT increased their robustness against errors to meet the societies demands of an uninterrupted supply stream. Especially, the supply chains for food products, like milk or dairy — products that are in need of cooling and special care during the whole process from 'farm-to-fork'[1013] — are critical for the safety of the population. In order to protect a flow of electronic documents that relate to the physical goods in production, storage and delivery the system used in ReSCUeIT facilitates digital signatures. ReSCUeIT used the term 'logical asset' to highlight that both need protection. A virtual asset is for example the electronic waybill that is linked to the delivery of the physical assets, e.g., a box of ice cream. The goal was to achieve a high probative value for the logical assets, while allowing certain blocks with sensitive information to be removed from the electronic documents. This was facilitated using an SSS. Further, the use of an SSS allowed to cryptographically lock the document at signature generation. After this, it was defined who shall or can change certain fields in forms, e.g. waybills. All deviations from this would become cryptographically detectable. This allowed supply chain partners to detect if there where any errors from the intended benign flow by verification of signatures at the input of the signed document.

Detecting deviations at any step immediately would allow to detect and react to errors, attacks on the underlying IT-Infrastructure, or potential sabotage of production. The research in ReSCUeIT was driven by potential real-world-scenarios constructed together with industry partners (REWE[1014], Eisbär Eis[1015], BaaM[1016], Dr. Oetker[1017], Kraftverkehr Nagel[1018]), SCM software developers (SAP AG[1019]), and universities (Universität Köln[1020], Universität Siegen[1021]).

Two contributions from this thesis made it into this system integration:

- the notion of integrity, and
- the increased probative value for SSS due to non-interactive public accountability.

The former enabled to capture existing demands and model the possibilities of subsequent edits together with authorizations that existed in the food supply chain workflow. The latter allowed to store the signed electronic documents as evidence records. Namely, the $pubacc\mathcal{SSS}$ scheme (see Sec. 13.4 and Sec. 13.5) was implemented as a set of micro web services. All algorithms from the SSS were implemented and made accessible to the workflow execution engine as web services. The XML transcoding was also implemented and each block from the $pubacc\mathcal{SSS}$ scheme corresponds to an XML node selected by dereferencing the node by `id` (see Sec. 18.1.3).

[1012] ReSCUeIT stands for 'Robustes und verfügbares Supply-Chain-Managment - Unterstützende IT-Plattform'
Project Duration: March 2010 - August 2013
Funding Agency: Federal Ministry of Education and Research (BMBF) & Joint French Consortium by ANR
Funding Number (FKZ): 13N10966
http://www.bmbf.de/pubRD/Projektumriss_RESCUE_IT.pdf

[1013] The term 'farm-to-fork' is used here to describe the whole process of a food product from production to consumer with all steps involved; see [2] for an example usage of the term.

[1014] REWE-Informations-Systeme GmbH

[1015] Eisbär Eis GmbH

[1016] BAAM Transporte + Lagerung

[1017] Dr. August Oetker AG

[1018] Kraftverkehr Nagel GmbH & Co KG

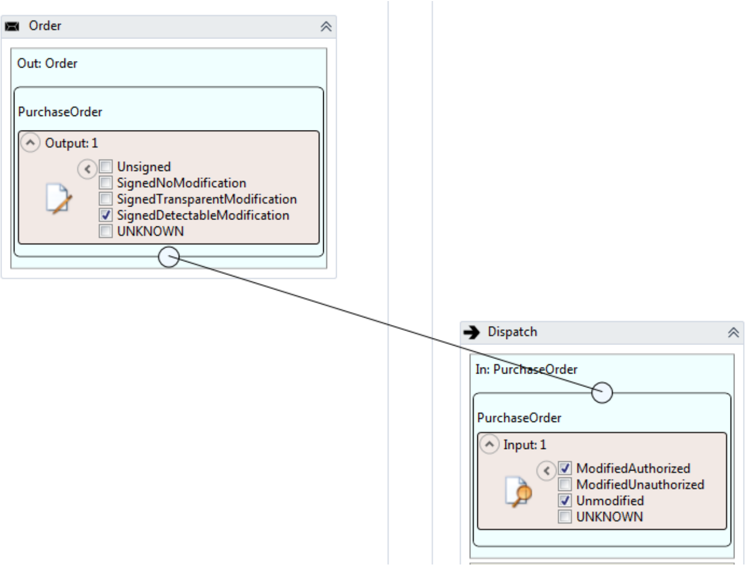
[1019] SAP AG, Abt. SAP Research

[1020] Universität zu Köln Seminar für Wirtschaftsinformatik und Informationsmanagement

[1021] Universität Siegen Institut für Wirtschaftsinformatik

18.1.1. The supply chain model with signature related control points

As an initial step the workflow is modelled by all workflow participants. The ReSCUeIT example workflow is the production, test, shipment and arrival of ice cream. The overall workflow spans four entities (REWE, BaaM, Eisbär Eis and Lab). Within ReSCUeIT SAP developed — among other things — the software with a graphical user interface (GUI) to model this workflow (see also *Monakova et al.* [352]). While the process sounds simple the overview in Fig. 135 shows that many assets are exchanged.



Order

Out: Order

PurchaseOrder

Output: 1

☐ Unsigned
☐ SignedNoModification
☐ SignedTransparentModification
☑ SignedDetectableModification
☐ UNKNOWN

Dispatch

In: PurchaseOrder

PurchaseOrder

Input: 1

☑ ModifiedAuthorized
☐ ModifiedUnauthorized
☑ Unmodified
☐ UNKNOWN

Figure 134. Screenshot of SAP's modelling GUI; an output control an SSS with PUB accountability (termed `SignedDetectableModification` in ReSCUeIT) should be applied to the asset `PurchaseOrder`; as input control the verification must succeed indicating the absence of unauthorized modifications; for more details on the user interface and the model refer to [352]

ReSCUeIT differentiates between actual physical assets, e.g., the ice cream, and logical assets, e.g., the electronic laboratory report.

The workflow model in ReSCUeIT aids the process designers by suggesting to specify input and output controls for each asset's exchange. For an logical electronic asset, like a waybill or a laboratory report, a digital signature can be inserted as an additional control. The output control applies a digital signature and the input control verifies its validity. Fig. 134 shows how the input and output controls get defined in the graphical user interface. The depicted example shows controls for the asset `PurchaseOrder` that is modelled to be signed by an SSS and the verification of the input control will only let it pass if no unauthorized subsequent modifications occurred. Being able to choose between SSS and CDSS integrity protection allows to specify what level of authenticity (origin authentication and integrity protection) is required to increase the robustness of the supply chain process. If `SignedNoModifications` had been selected on the output control, a CDSS would be used to generate the signature. By the selection of accepted verification results the modeller will select a conforming MSS: In this example of the input control's configuration the digital signature scheme for signing must be either a CDSS with an output control of `SignedNoModifications` as this allows to meet `Unmodified`, or an MSS which offers ≥1CD integrity in order to meet to detect `Modifiedauthorized`. The case `ModifiedUnauthorized` found as an input control in Fig. 134 is to model a reaction when receiving invalid signatures.

In the ReSCUeIT example workflow, the first step is to order ice cream. Once the production is ready, the following two processes are carried out in parallel: The delivery on trucks to the destination happens while a sample from the shipped batch of ice cream is sent to a trustworthy independent food laboratory for product testing. In case the batch is tested to be of low quality, additional mechanisms from ReSCUeIT will allow to recall the delivery of the ice cream, or to ensure that already delivered ice cream will not be in store shelves or sold until the tests are finished and indicate that this is a high quality food product. This quality control and fully integrated recall process made the food supply chains executed in ReSCUeIT more robust. For SSS it is important that analysis showed that the report generated by the food laboratory in the real world scenario showed information that were not needed for all supply chain partners. Hence, the report in XML was marked as in need of a signature that allowed to be sanitized.

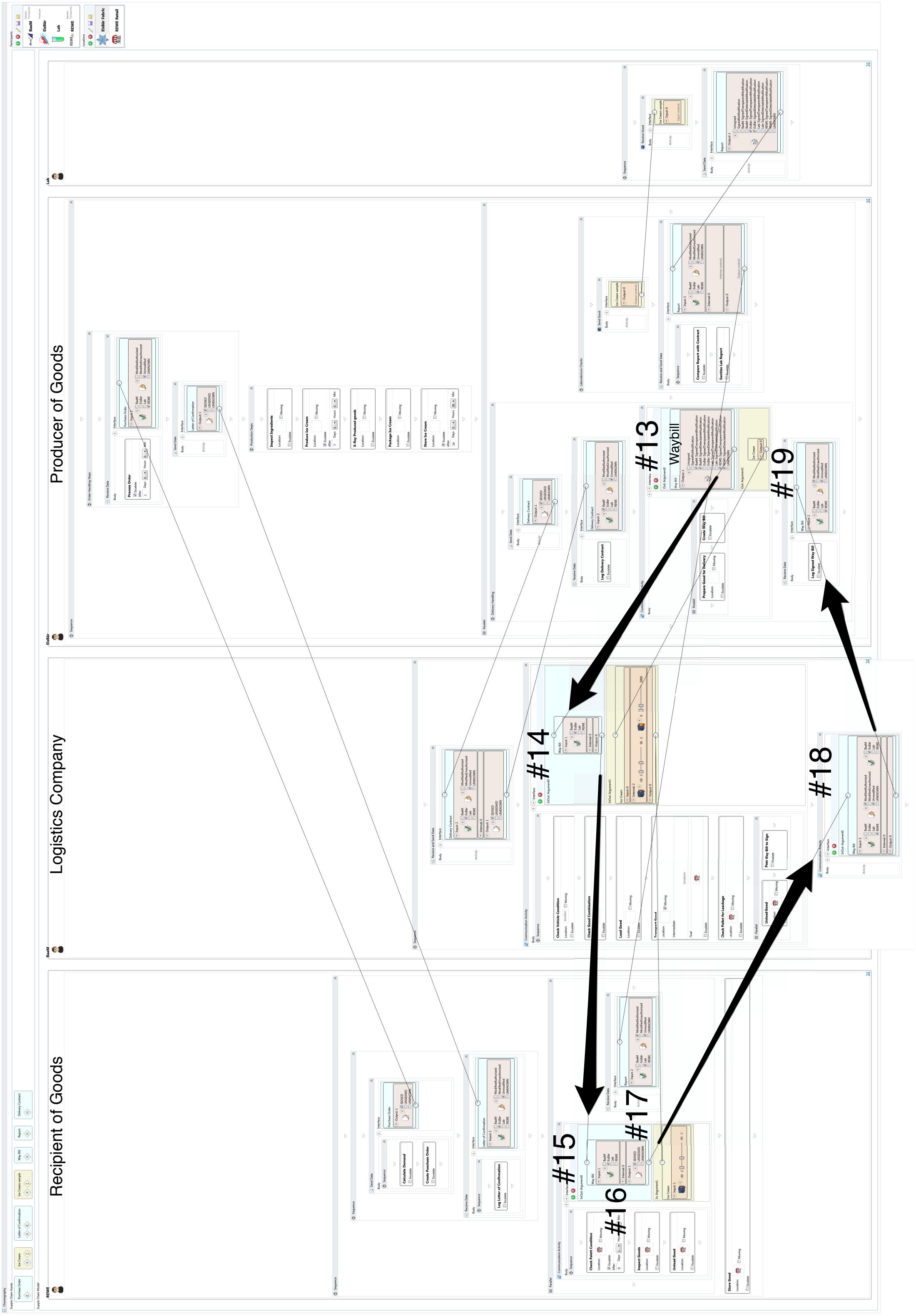


Figure 135. ReSCUeIT's example food supply chain with four partners (from left/bottom to right/top: receiver, logistics, producer and food laboratory); the logical assets involve, among others, a sanitizable signed waybill to allow modification of the actually delivered amount in one specific field and a redactably signed laboratory report to allow removing personal data. Highlighted is the flow of the waybill; for more details on the user interface and the model refer to [352]

In the example workflow this is modelled by selecting a sanitizable signature generation as the `controlPoint="Output"` for the `controlledAsset="LabReport"` is set to `Changeable Signature` in the model. See Listing 1. for an XML representation of another asset of the modelled workflow, the protection of the waybill. Here, a sanitizable signature generation is designed by the controlPoint="Output" for the controlledAsset="WayBill", that is codified by the control being Changeable Signature in the modelled workflow. The control includes the specification of the dedicated Sanitizer (here REWE) and the properties of the sanitizable signature scheme to support SignedDetectableModification using the <ValidStates>-element. In the example workflow from ReSCUeIT, the business process at the participant Eisbaer will, following what is modelled, generate a signature that can be subsequently modified only by the party REWE. This is implemented by facilitating a suitable SSS for the actual generation of the signature. ReSCUeIT's example workflow then has input controls on subsequent workflow steps which require that the verification must succeed to enforce the absence of unauthorized modifications as depicted in Listing 1.. This ensures that only authorized subsequent modification are done which must also adapt the SSS signature in order for it to remain valid on the modified content and this needs the knowledge of REWE's corresponding secret sanitization key.

Listing 1. XML from ReSCUeIT-Modeller GUI developed by SAP showing the request to apply an SSS (indicated by Changeable Signature) to the logical asset WayBill

```
<Interface><Output id="Argument_14" Asset="WayBill"/></Interface>
<Controls>
  <StateControl name="Changeable Signature" controlPoint="Input" controlledAsset="
      WayBill">
     <ValidStates> ...
          <State>REWE: SignedDetectableModification</State> ...
     </ValidStates> ...
  </StateControl>
</Controls> ...
```

18.1.2. State of signatures

Additional to the detection whether or not the signature protecting the logical asset is valid, ReSCUeIT required to identify the signature method used, i.e., what level of integrity protection the document was given. In ReSCUeIT the extended notion of integrity[1022] enabled distinguishing between three different classes for a signature scheme with the following identifiers:

- SignedWithNoModifications,
- SignedWithAuthorizedTransparentModifications,
- SignedWithDetectableModifications.

While the first state identifies a CDSS with NCA – 1CD integrity protection, the latter two would indicate the usage of an MSS with ACA integrity protection. The second would identify a transparent scheme, i.e., non-public interactively accountable scheme that would give only ACA – ≥1CD – INT integrity protection. Only the latter would give detectable, i.e. public, accountability and thus stand for schemes offering ACA – ≥1CD – PUB integrity protection.

During verification of a signed ReSCUeIT-asset the ReSCUeIT system will detect if the signed asset was modified, since the integrity is protected either through the $pubacc\mathcal{SSS}$ or a standard signature scheme. To clearly identify in which state a signed asset is, the following steps are done:

1. Identify if the correct scheme for generation of the signature was used using the function IdentifySignatureScheme().
2. Verify the actual signature against the expected sender's public key using the function Verify().
3. Detect the presence of authorized subsequent modifications, i.e., check if the document was modified or not using the Judge() function.

Thus, depending on the signature scheme ReSCUeIT differentiated different verification results. In more detail the Validate() function — a combination of different MSS algorithms, i.e. Verify() and Judge() and a helper function IdentifySignatureScheme() — generates its output as follows:

- UNMODIFIED
  (Verify()=VALID and
  IdentifySignatureScheme()=SignedWithNoModifications)

[1022] See Chapter 6 for the extended definition of integrity from this thesis.

- UNMODIFIED
  (Verify()=VALID and
  IdentifySignatureScheme()=SignedWithDetectableModifications
  and Judge()=Signer)
- POSSIBLY_MODIFIED_AUTHORIZED
  (Verify()=VALID and
  IdentifySignatureScheme()=SignedWithAuthorizedTransparentModifications)
- MODIFIED_AUTHORIZED
  (Verify()=VALID and
  IdentifySignatureScheme()=SignedWithAuthorizedDetectableModifications
  and Judge()=Sanitizer)
- MODIFIED_UNAUTHORIZED
  (Verify()=INVALID)
- UNKNOWN
  (else)

Note, when facilitating an SSS with public non-interactive accountable (PUB) it is possible to allow for a non-interactive differentiation between the outcome of the second and third state. Only then the algorithm denoted Judge() can generate useful output without additional interactions. Obviously the scenarios used within ReSCUeIT demanded a sufficient level of detection. As a result, ReSCUeIT required a mechanism with at least $\mathrm{ACA}-1\mathrm{CD}-\mathrm{PUB}$ integrity protection to identify for an SSS signature whether the Signer or the Sanitizer is accountable for the message. Differentiating between MODIFIED_AUTHORIZED and MODIFIED_UNAUTHORIZED is hence possible due to the usage of the *pubaccSSS* scheme that was implemented for ReSCUeIT (see Sec. 13.5) which offers the required integrity protection (see Sec. 13.5.3).

18.1.3. SSS as XML DSIG compatible signature

ReSCUeIT identified the need to secure logical assets in the XML format within the supply chain. This required to map the SSS and RSS signatures that work on blocks onto the XML of the document. Further, ReSCUeIT demonstrated that the resulting signature can be stored inside the XML such that the signed XML stays compliant with the XML DSIG specification [26]. In a nutshell, a node from the XML document is mapped into a block for the SSS. Listing 2. reproduces the information contained in the SOAP request in ReSCUeIT to request the signature generation for a XML document. For brevity, certain sections have been shortened using '...' as a placeholder[1023]. The sanitizable signature scheme is identified by <sig:signatureMethod>http:www.example.orgxmldsig-more#bps</sig:signatureMethod> in line 27. The information that is protected by the signature is encapsulated in the <sig:sInfo> element, as found in line 4 of Listing 2. The blocks in the ReSCUeIT document that shall remain unmodifiable in subsequent steps are the following ones identified by their XML identifier (id):

- items which contains the information what product was tested,
- 12-04168-001-2 which contains the information about the tests, as well as
- alltestresults and
- note which contain the final result.

Those blocks are listed as uri references. For example, line 13 refers to alltestresults and line 22 is referring to the XML element with the id 12-04168-001-2. The remaining content pointed at by lines 35–37 has been removed from Listing 2. for readability. The admissible blocks shall become changeable by the entity named eisbaer. Line 41 identifies the Sanitizer as

[1023] For example, line 3 of Listing 2. usually contains all the XML content that becomes integrity-protected by a sanitizable signature when it is selected by an xpointer expression. An example for the content, e.g., a laboratory report, is given in Listing 3.

<sig:sanID>eisbaer</sig:sanID>. During signature creation the Signer is specified by providing the ReSCUeIT web service with an entity identifier and a passphrase. The communication with the trusted server that ran the signature generation web service was secured by transport layer security (TLS) within ReSCUeIT.

A privacy risk assessment for the lab report used in ReSCUeIT identified several elements to carry information that could be subject to later subsequent removal due to containing personal information or trade secrets. Namely, the following parts of the XML document are identified as sanitizable and get thus referenced in the element <sig:admReferences> inside <sig:xmlObject>, which is XML DSIG's generic object element: labname, labpersonell, labadress, clientname, client personell, clientadress, and producer. Again, the blocks are represented as XML selectors, e.g., line 36 <sig:uri>#xpointer(id('labname'))</sig:uri> selects data containing the information about the laboratory. As it is listed as admissible, it is considered subsequently removable to allow sanitizations to protect trade secrets or personal data.

Listing 2. SOAP request within ReSCUeIT to sign the provided XML that represents the food laboratories report on the chemical and biological contamination levels

```
1   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><
        soapenv:Body>
2   <sig:SignatureController xmlns:sig="http://signatureController.sichere-warenketten.de">
3   <sig:XMLAsString> ... </sig:XMLAsString>
4   <sig:sInfo>
5     <sig:canonicalizationMethod>http://www.w3.org/2001/10/xml-exc-c14n#</
          sig:canonicalizationMethod>
6     <sig:references>
7         <sig:digestMethod>
8           <sig:algorithm>http://www.w3.org/2001/04/xmlenc#sha256</sig:algorithm>
9         </sig:digestMethod>
10        <sig:transforms>
11          <sig:transformAlgorithm>http://www.w3.org/2000/09/xmldsig#enveloped-signature</
                sig:transformAlgorithm>
12        </sig:transforms>
13        <sig:uri>#xpointer(id('alltestresults'))</sig:uri>
14    </sig:references>
15    <sig:references>
16        <sig:digestMethod>
17          <sig:algorithm>http://www.w3.org/2001/04/xmlenc#sha256</sig:algorithm>
18        </sig:digestMethod>
19        <sig:transforms>
20          <sig:transformAlgorithm>http://www.w3.org/2000/09/xmldsig#enveloped-signature</
                sig:transformAlgorithm>
21        </sig:transforms>
22        <sig:uri>#xpointer(id('12-04168-001-2'))</sig:uri>
23    </sig:references>
24    <sig:references>
25        ...
26    </sig:references>
27    <sig:signatureMethod>http://www.example.org/xmldsig-more#bps</sig:signatureMethod>
28  </sig:sInfo>
29  <sig:keyInfo>
30    <sig:KeyName>laboratory</sig:keyName>
31        <sig:pw>labsSecretPIN</sig:pw>
32  </sig:keyInfo>
33        ...
34  <sig:xmlObject>
35        <sig:admReferences id="admissibleBlocks">
36          <URI>#xpointer(id('labname'))</URI>
37          <URI>#xpointer(id('labpersonell'))</URI>
38          <URI>#xpointer(id('labadress'))</URI>
39          ...
40        </sig:admReferences>
41        <sig:PKsan id="pksan" sanID="eisbaer">MMsdr24SFS...ghSd3dfn3=</sig:PKsan>
42  </sig:xmlObject>
43  </sig:SignatureController></soapenv:Body></soapenv:Envelope>
```

Those XML structures are compliant to XML DSIG [26]; the early result that this type of integration of SSS into XML is indeed feasible was published as joint work with *K. Samelin* and *J. Posegga* [391] (see Appendix A publication n° 6).

Listing 3. XML-File Example of a Laboratory Report

```
...
<Test id="12-04168-001-2">
      <TestedProduct>001: - Rewe, Erdbeer Fruchteis 900ml,
                  Mat.-Nr.: 05615, 06:39, Herst.-datum: 08.10.2012
      </TestedProduct>
      <TestSampleReceived>10.05.11</TestSampleReceived>
      <TestedChargenNummer>L/LEE 1124 0018</TestedChargenNummer>
      <TestDate end="21.10.2012" start="20.10.2011"/>
      <TestFor>Salmonella spp. / 25 g</TestFor>
      <TestResult unit="per 25 g">negativ</TestResult>
...
</Test>
...
<Result id="alltestsresult">001: verkehrsfaehig</Result>
<ResultNote id="note">
       Aufgrund der ermittelten Ergebnisse aus den hier durchgefuehrten Untersuchungen
            ergeben sich
       bezueglich der Produktbeschaffenheit keine Anhaltspunkte fuer eine
           lebensmittelrechtliche
       Beanstandung. Somit ist die Probe als verkehrsfaehig zu bewerten.
</ResultNote>
...
<Tester>
      <Name id="labname">Lebensmittelanalytik GmbH</Name>
      <Personell id="labpersonell">Max Mustermann</Personell>
      <Address id="labadress">
        <Street>Testerweg 13</Street> <City>Wolfrathshausen</City>
        <Country>Germany</Country> <PostalCode>12345</PostalCode>
      </Address>
</Tester>
...
```

18.1.4. Waybill protection and subsequent authorized modification within ReSCUeIT

The workflow depicted in Fig. 135 on page 492 involves also the waybill as an important document in supply chains. Examples of such a waybill have been depicted in this thesis in the introduction to motivate the use of sanitizable signature schemes, e.g., Fig. 3 on page 8. An initial step within the example workflow from ReSCUeIT is the creation of the logical asset waybill. In ReSCUeIT the waybill conforms to the DESADV edifact message [483]. The waybill is traversing from its origin party Eisbaer, were it is initially getting signed (Step 13), over to the logistics partner Baam (Step 14) for delivery and finally it arrives at REWE (Step 15). It will then be potentially modified by REWE in an authorized manner (Step 16) and then signed unchangeably by REWE (Step 17). Then it starts it traversal back to the originator: The waybill traverses via the entity of Baam (Step 18) back to Eisbaer (Step 19). By this process, the waybill covers the complete delivery process and serves as evidence of receipt to Eisbaer and allows Baam to have a a signed proof of delivery and a signed proof of being contracted with the shipment. To provide this evidence record the waybill must not be subsequently modified in unauthorized ways and all authorized modifications must have been carried out only by the authorized partners, i.e. only the recipient of goods will be able to modify a special field of the waybill to record deviations. Hence, the first step (Step 13 in the original workflow) is to generate a sanitizable signature that offers the correct properties.

18.1.4.1. Generating a sanitizable signature on the waybill (Step 13)

Listing 1. showed the XML of the modelled control point. Also for the DESADV document — the waybill — the XML is divided into fixed and admissible blocks by selecting nodes from the XML by their `id`-attribute.

Selecting nodes by `id`-attribute is discouraged as it increases the potential to fall victim to XML wrapping attacks. This thesis is aware of potential wrapping attacks, e.g. [200, 333, 449]. As a mitigation the XML content that corresponds to the block should be selected by giving full paths starting at the document's root.

Nearly all XML elements of the DESADV XML document were identified in ReSCUeIT as fixed blocks, i.e. unchangeable. This is codified by specifying the full XML without the signature and without the admissible blocks as fixed using `URI=""` in line 66. This reference initially selects the complete contents of the XML document and then carves out, using two XPath transforms in line 70 and 73, first the signature element (`not(ancestor-or-self::sig:Signature)`) and second the element selected to be admissible for a subsequent modification (`not(ancestor-or-self::*[@id='CPS1discrepancy'])`). This way the important unchangeable information of the waybill is protected against any undetected authorized or unauthorized subsequent modification. For example `BGM` contains the document type, `SG2` contains the address, `SG9` contains handling instructions, and `LIN-4388816187235` contains information about the product and the to be delivered quantity (`quantity="200"` in line 54). There is only one admissible element inside the DESADV which shall stay subsequently modifiable by `REWE`. The potential modification accounts for the possibility of REWE to identify any discrepancies between the ordered amount of ice cream and the delivered amount. It is, as described in Sec. 18.1.3, specified within the list of admissible blocks in `<sig:admReferences>` within the XML DSIG compatible `<sig:xmlObject>`. In line 108 it is referenced by its `id`-attribute's value of `CPS1discrepancy`. The XML element itself is in lines 39–42. The resulting signed waybill with the full contents of an DESADV in XML is too big to reproduce here. Listing 4. gives a shortened version to still give an impression on how the multiple blocks fixed and admissible) is integrated into XML DSIG to encode the sanitizable signature scheme *pubacc*$\mathcal{SSS}$ identified with `xmldsig-more#bps`. Also note that in this example XML the fixation of the ADM as well as the endorsement and fixation of the Sanitizer becomes visible: Both are elements of the XML and both get signed using the same CDSS that is used for the fixed parts as it is listed as a reference element in line 79 and 86.

Listing 4. XML of signed unmodified DESADV (waybill) message; shortened and condensed for readability; original from ReSCUeIT

```
1   <DESADV xmlns="http://smooks.org/UNEDI/D09BUN/DESADV" id="WayBill">
2         <BGM id="BGM">
3               <documentMessageName documentName="Way Bill" documentNameCode="DESADV"/>
4               <documentMessageIdentification/>
5         </BGM> ...
6         <segmentGroup1 id="SG1">
7               <RFF id="RFF">
8                     <reference referenceCodeQualifier="ID" referenceIdentifier="
                          WayBill1234242"/>
9               </RFF> ...
10        </segmentGroup1>
11        <segmentGroup2 id="SG2"><NAD cityName="Appensen" countryIdentifier="de" id="NAD"
              postalIdentificationCode="21641"> ... <partyName partyName="EISBAER"/><
              street streetAndNumberOrPostOfficeBoxIdentifier="Eisbaerstrasse 1"/> ... </
              NAD><LOC id="LOC" locationFunctionCodeQualifier="LOC"><
              locationIdentification ... locationIdentifier="PICKUP" locationName="EISBAER
              "/></LOC>
12   ...
13        <segmentGroup5 id="SG5">
14              <TOD id="TOD" ...>
15                    <termsOfDeliveryOrTransport codeListIdentificationCode="TOD12445"
                          codeListResponsibleAgencyCode="DE"
                          deliveryOrTransportTermsDescription="Keep the good below MINUS
                          16 degree celsius" deliveryOrTransportTermsDescriptionCode="
                          TMPCTL"/>
16              </TOD>
17              <LOC id="LOC-1" locationFunctionCodeQualifier="DROP-LOCATION-1">
18                    <locationIdentification locationIdentifier="LOC1234" locationName="
                          REWE Store Koeln"/>
19                    ...
20              </LOC>
21              <LOC id="LOC-2" locationFunctionCodeQualifier="DROP-LOCATION-2">
22                    <locationIdentification locationIdentifier="LOC1235" locationName="
                          REWE Markt Vierlindenmarkt"/>...
23              </LOC>
24              <FTX id="SG2-FTX" languageNameCode="en" textSubjectCodeQualifier="Route
                    Instructions">
25                    <textLiteral freeText="Please be at LOC-1 before 12:00 and at LOC-2
                          no later than 16:00."/>
26              </FTX>
27
28   ...
29        <segmentGroup9 id="SG9">
30              <HAN id="HAN">
31                    <handlingInstructions codeListIdentificationCode="12344-2"
                          codeListResponsibleAgencyCode="?" handlingInstructionDescription
                          ="Keep the good below MINUS 16 degree celsius"
                          handlingInstructionDescriptionCode="KeepFroozen"/>
32              </HAN>
33              <TMP temperatureTypeCodeQualifier="KeepFrozenTemp">
34                 <temperatureSetting measurementUnitCode="Celsius" temperatureDegree="-18
                       "/>
35              </TMP>
36   ...
37        <segmentGroup10 id="SG10-PackagingDescription">
38   ...
```

```
        <QVR discrepancyNatureIdentificationCode="-" id="CPS1discrepancy">
          <quantityDifferenceInformation ... varianceQuantity="-"/>
          <reasonForChange changeReasonDescription="-"/>
        </QVR>
  ...
      <segmentGroup17 id="LIN-4388816187235">
        <LIN lineItemIdentifier="4388816187235"/>
        <PIA productIdentifierCodeQualifier="PIA">
            <itemNumberIdentification itemIdentifier="ArtNr 338589"/>
        </PIA>
        <IMD descriptionFormatCode="TXT">
            <itemCharacteristic codeListIdentificationCode="DE-FROZEN" codeListResponsibleAgencyCode="EISBAER" itemCharacteristicCode="FROZEN"/>
            <itemDescription codeListResponsibleAgencyCode="EISBAER" itemDescription="REWE Bourbon Vanille, 500ml" itemDescriptionCode="TXT" languageNameCode="en"/>
        </IMD>
        <QTY>
            <quantityDetails measurementUnitCode="PCE" quantity="200"/>
        </QTY>
  ...
        <FTX languageNameCode="en" textSubjectCodeQualifier="TXT">
            <textLiteral freeText="REWE Beste Wahl Bourbon Vanille Eis Art.-Nr. 338589"/>
        </FTX>
  ...
      <sig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
        <sig:SignedInfo>
         <sig:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
         <sig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
         ...
         <sig:Reference URI="" Id="allFixedBlocks">
             <sig:Transforms>
               <sig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
               <sig:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
                <sig:XPath>not(ancestor-or-self::sig:Signature)</sig:XPath>
               </sig:Transform>
               <sig:Transform Algorithm="http://www.w3.org/TR/1999/REC-xpath-19991116">
                <sig:XPath>not(ancestor-or-self::*[@id='CPS1discrepancy'])</sig:XPath>
               </sig:Transform>
             </sig:Transforms>
             <sig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
             <sig:DigestValue>1VxZmzsDP4T97...xSMTjiRd10=</sig:DigestValue>
         </sig:Reference>
         <sig:Reference URI="#xpointer(id('pksan'))" Id="fixPksan">
             <sig:Transforms>
               <sig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
             </sig:Transforms>
             <sig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
             <sig:DigestValue>sYotTbMHbb8UM...cHSxJCR+8s=</sig:DigestValue>
         </sig:Reference>
         <sig:Reference URI="#xpointer(id('admissibleBlocks'))" Id="fixADM">
             <sig:Transforms>
               <sig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
             </sig:Transforms>
             <sig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
             <sig:DigestValue>duJe3f++7d+ZQ...zOat6ToxAc=</sig:DigestValue>
         </sig:Reference>
        </sig:SignedInfo>
        <sig:SignatureValue>lCDlzIVc6hZ...BJPQ=</sig:SignatureValue>
        <sig:KeyInfo>
            <sig:KeyValue>
             <sig:RSAKeyValue>
                  <sig:Modulus>1A2stqJi9O46Tg...R51aJii7qC4tU=</sig:Modulus>
                  <sig:Exponent>AQAB</sig:Exponent>
             </sig:RSAKeyValue>
            </sig:KeyValue>
            <sig:X509Certificate>MIIGE5/tPzOssfsS343fsSFJLe4xS6TC...q5/tPSgszOdsfSFSFF6Tpsry+iers</sig:X509Certificate>
        </sig:KeyInfo>
        <sig:Object>
            <sig:signatureMethod>http://www.example.org/xmldsig-more#bps</sig:signatureMethod>
            <sig:PKsan id="pksan" sanID="rewe">rOtABysf23A...AZNAFJMSUM=</sig:PKsan>
            <sig:admReferences id="admissibleBlocks">
             <URI>#xpointer(id('CPS1discrepancy'))</URI>
            </sig:admReferences>
        </sig:Object>
      </sig:Signature>
</DESADV>
```

18.1.4.2. Verification of the SSS signature on the waybill by the delivery company (Step 14) and the recipient of goods (Step 15)

As the waybill was signed at its origin — Eisbaer — (Step 13) it can be verified subsequently by Baam (Step 14) and and forwarded. Namely, the workflow assumes that the signed waybill has arrived at REWE alongside the goods (Step 15). Before handling physical assets the digital logical asset, the signed waybill, is verified inside the ReSCUeIT flow. The signature in this case was generated with the SSS. It gets verified before the waybill enters into the workflow at the delivery company. This is modelled by an input control. If the signature is valid then no problems are detected regarding the logical asset and the delivery process continues.

Once the goods mentioned in the electronic waybill are received, the goods' recipient checks if the signature is still valid. In this example workflow the recipient of the goods and the signed document is REWE. This is again modelled by an input control. A positive authentication of origin, which is inherent to the facilitated SSS, assures REWE that the waybill has not changed in an unauthorized manner since it originated from the product's producer. The sanitizable signature scheme of $pubacc\mathcal{SSS}$ used for this in ReSCUeIT offers non-interactive public (PUB) accountability in order to let REWE identify for sure that is was still unmodified and not yet subject to any (authorized or not) subsequent modification.

18.1.4.3. Potential sanitization of the SSS signature on the waybill by recipient of goods (Step 16)

In Step 16 REWE carries out a possible sanitization in a field of the waybill that the Signer Eisbaer specified for modification by REWE: `CPS1discrepancy` can be edited to indicate if there is a discrepancy between the real physically delivered goods and the waybill's logical description of them, e.g., in quantity. Step 16 is the fourth of several steps handling the logical asset waybill. REWE has been authorized by the Signer to do this subsequent modification of the waybill. The policy that only REWE is able to do this modification was cryptographically encoded, as REWE's secret sanitization key must be involved. This got codified when the document got signed by the product's producer `eisbaer`, who authorized this sanitization in the signing step described earlier in Sec. 18.1.4.1 as Step 13. In the XML from Listing 4., this delegation is visible as the Sanitizer's public key of REWE is included in the XML in line 105 and is part of the XML that is signed using a CDSS, as indicated by the reference in line 79 pointing to the element with the identifier `'pksan'`.

ReSCUeIT's example assumed that the previously empty field `CPS1discrepancy` is sanitized to look somewhat like the excerpt shown from Listing 5.

Listing 5. Sanitized portion of the DESADV node QVR

```
<QVR discrepancyNatureIdentificationCode="DME" id="CPS1discrepancy">
 <quantityDifferenceInformation quantityTypeCodeQualifier="PCE" varianceQuantity="-20"/
      >
 <reasonForChange changeReasonDescription="20 packages damaged - lid was open"/>
</QVR>
```

This indicates that 20 pieces of the to be delivered 200 pieces (PCE) are damaged. REWE, as the dedicated Sanitizer, will then be able to potentially modify the waybill in an authorized manner (Step 16) and then sign the potentially modified waybill unchangeably (Step 17). After this, the modified and countersigned waybill (Step 17) is traversing back via Baam (Step 18) to Eisbaer (Step 19).

18.1.4.4. Standard signature on the waybill by recipient of goods (Step 17)

An authorized Sanitizer can produce several documents or can do several modifications in sequence. However, none of the resulting validly signed sanitized documents can be ordered in time. To prohibit the Sanitizer from cheating and repudiating later that the Sanitizer indeed wrote "20 packages damaged - lid was open", the example workflow contains controls that additionally generate a signature by the recipient over the waybill's field `CPS1discrepancy` with a signature scheme that offers NCA – PUB integrity, i.e., the signature disapproves subsequent modifications (NCA) and provides non-repudiation, e.g., using a standard CDSS. This signature is generated regardless of an actual sanitization of the contents of the field `CPS1discrepancy` in order to allow the supply chain partners to see that the recipient of the goods has actually received the electronic document of the waybill. The reason is that this logical asset represents the physical asset: A countersigned waybill indicates that the physical goods were received. The resulting standard signature is also conforming to the XML DSIG standard and is shown in Listing 6..

Listing 6. XML part with the standard CDSS that REWE adds over the sanitized version to document that it will not do further subsequent edits to the `DESADV` node with the id `CPS1discrepancy` which stays sanitizable for REWE

```
<sig:Signature xmlns:sig="http://www.w3.org/2000/09/xmldsig#" Id="fixSanitization">
  <sig:SignedInfo>
    <sig:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
      <sig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <sig:Reference URI="#xpointer(id('CPS1discrepancy'))">
        <sig:Transforms>
          <sig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
              signature"/>
        </sig:Transforms>
        <sig:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
        <sig:DigestValue>jpioltZI6WzfY...uW47P7/a6WqoxE=</sig:DigestValue>
      </sig:Reference>
  </sig:SignedInfo>
  <sig:SignatureValue>tyjAXNCEizIMPe...2TTIHQ=</sig:SignatureValue>
  <sig:KeyInfo>
    <sig:KeyValue><sig:RSAKeyValue>
      <sig:Modulus>1A2stqJi...qC4tU=</sig:Modulus>
      <sig:Exponent>AQAB</sig:Exponent>
    </sig:RSAKeyValue></sig:KeyValue>
  </sig:KeyInfo>
</sig:Signature>
```

18.1.4.5. Verification of both signatures on the waybill by sender of goods (Step 18 a&b)

As the final step in the circulation of the waybill, it will arrive as a proof of delivery at the sender, e.g., Eisbaer. This party, like any other party can now verify both signatures. This is possible because the *pubacc*$\mathcal{SSS}$ offers non-interactive public accountability (PUB). A positive verification of the *pubacc*$\mathcal{SSS}$ signature will yield that Eisbaer was the source of the waybill and that REWE was acting as a Sanitizer authorized by Eisbaer when it added information about damaged products into the waybill's field `CPS1discrepancy`. A positive verification of the standard signature scheme will yield that REWE will no longer be able to edit the authorized sanitized entry regarding the damaged 20 units.

18.1.5. Implementation of *pubacc*$\mathcal{SSS}$ as web services for generation, verification, sanitization of signatures in XML within ReSCUeIT

Within the ReSCUeIT project an implementation of all algorithms regarding MSS signatures was provided[1024]. The functionality has been implemented as a set of web services and the extended functionalities offered by MSS are used for example to achieve the

`Modified` property. As shown in Fig. 134, the current state of modification is helpful and is a result of a signature verification if the scheme used has the right properties: This requires a non-interactive accountability and a trade secret preserving modification. The scheme *pubacc*$\mathcal{SSS}$ (see Sec. 13.4 and Sec. 13.5 for design and proofs) obtained as a result of this thesis' work was chosen for implementation because it achieves the non-interactive public (PUB) accountability needed for ReSCUeIT. In detail, it does not require the Verifier to do online status checks. Under the assumption that the verifying party has a trusted and hence valid public key of the Signer it can verify signatures offline. In ReSCUeIT a correct key distribution must happen during the setup of the collaborative supply chain.

The implementation was done in Java and in Ruby On Rails[1025]. All functionality was exposed as web services with standard and open REST APIs. They were offered on the TomCat server from the *Apache* Foundation[1026].

[1024] For more information there are videos of the demo available; for the generation of a sanitizable signature on a laboratory report (as described in Listing 3.) in ReSCUeIT see `http://henrich.poehls.com/papers/video_labreport.mp4` [last accessed: Jul. 2017] and for a standard signature on the purchase order see `http://henrich.poehls.com/papers/video_purchaseorder.mp4` [last accessed: Jul. 2017].

[1025] See `rubyonrails.org` [last accessed: Jul. 2017] for details.

[1026] See `tomcat.apache.org` [last accessed: Jul. 2017] for details.

Operation	**Number of XML references (block)**	**Speed (ms)**
$pubacc\mathcal{SSS}$ Signing (Step 13)	15	846
$pubacc\mathcal{SSS}$ Verification (Step 14)	15	409
$pubacc\mathcal{SSS}$ Sanitization (Step 16)	1	1205
Standard Signature Generation (Step 17)	1	778
Standard Signature Verification (Step 18a)	1	238

Table 33. Total speed in milliseconds (ms) of the different operations carried out on a waybill file used in ReSCUeIT

18.1.5.1. Speed and performance of standard and $pubacc\mathcal{SSS}$ signatures

The speed for sanitizable signatures (generation, verification and sanitization) are for the ReSCUeIT implementation that applied the $pubacc\mathcal{SSS}$ or standard signatures on XML files. The algorithm itself is described in Sections 13.4 and 13.5 and prototypical performance was also measured (see Sec. 13.5.2). These results also got published as joint work with *C. Brzuska* and *K. Samelin* in the 9th European PKI Workshop: Research and Applications (EuroPKI) in 2012 [68] (see Appendix A publication nº 11).

Tab. 33 shows the speed in milliseconds (ms) for signature generation and verification of all 15 references that are contained in the ReSCUeIT example. As the underlying signature scheme ReSCUeIT chose the signature to be RSA-PSS with a key size of 2048 Bit and SHA256 as the standard hash algorithm. The standard signature that was added in Step 17 and 18a is faster, compared to $pubacc\mathcal{SSS}$, because it requires far less XML reference extractions and hashes, and only one signature operation instead of two. Fig. 136 offers a visual comparison of the duration in milliseconds (ms) for the complete operation carried out on the XML file in the different steps of the ReSCUeIT workflow depicted in Fig. 135 on page 492.

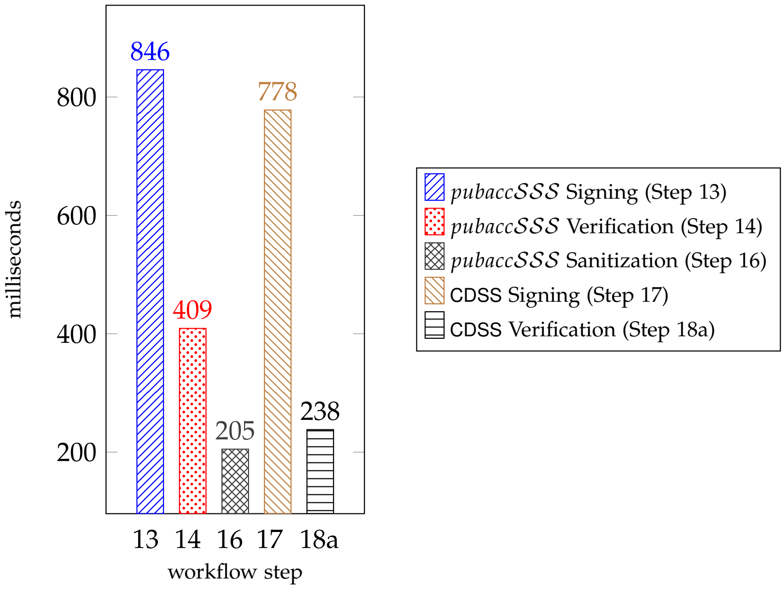


Figure 136. Runtime in milliseconds (ms) of the different operations carried out on a waybill file by the web services implemented in ReSCUeIT

18.2. Proof of concept: Information source authentication and integrity preserving application of privacy-enhancing technology (PET) in the Internet-Of-Things (IoT) for healthcare

These results have been published in the joint work with *C. Frädrich, W. Popp, N. Rakotondravony* and *K. Samelin* at ICICS 2016 [198] (see Appendix A publication nº 53).

This section describes the application of an RSS on an IoT device in the scenario where the device collects trustworthy health data but the patient still wants to control the granularity and level of detail before forwarding authenticity-protected data to any third parties.

18.2.1. RSS on IoT devices gathering healthcare data

An RSS allows the redaction of blocks from signed data. The scheme presented as $merge\mathcal{RSS}$ (see Sec. 14.11 and Sec. 14.12) allows via updates that the signatory can add new elements, while signatures can be merged by third parties under certain conditions.

The joint work [198] proposes a framework for two new real-life application scenarios and implements it using the secure RSS with added functionality on three different platforms, ranging from a potent cloud to a very resource-constrained Android device. The evaluation shows impractical run time especially on the IoT device for the existing construction that was proven to be secure in the standard model. However, an adjusted scheme, which was proven to be secure in the random oracle model (ROM), is presented which offers far better performance[1027].

For example take the privacy and authenticity preserving data workflow sketched in Fig. 137.

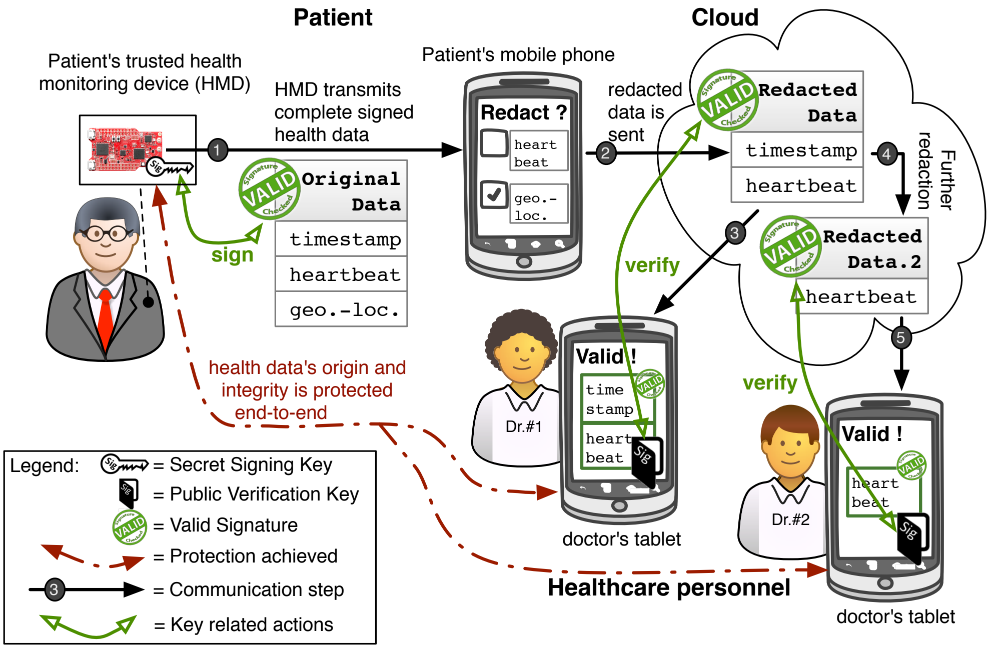


Figure 137. Example workflow for e-health use-case: integrity and origin of remaining data remains verifiable and protected against active attacks; redaction needs no keys; as published jointly by *Frädrich, Pöhls, Popp, Rakotondravony, and Samelin* [198]

The evaluation of the actual performance was done on three very different environments: a constrained IoT device, a standard Android smart phone, and a powerful computing device as found in clouds. Interestingly, the evaluation shows that the approach is suitable for workflows involving cloud as well as the Android mobile phone under consideration, if implemented with care and certain optimisations, including some changes to the used RSS. Namely, the original $merge\mathcal{RSS}$ construction from [387] — cryptographically secure in the standard model — requires extensive primality tests. For the sake of reduced runtime the implementation reduced the cryptographic strength to the practically still useful random oracle model (ROM) to obtain usable and for many workflows satisfying performance. In this section the security proof of the altered construction is omitted, it can be found in [198].

In Fig. 137, healthcare data is gathered by a constrained, Internet-of-Things (IoT) like, yet certified and trusted, medical health monitoring device (HMD). The goal is to allowing to share the resulting data with authenticity guarantees, e.g. that data was not maliciously modified and originated from a certain HMD. However, at the time of data generation and application of the integrity and authenticity protection, one

[1027] This was expected when switching from security in the standard model to security in the random oracle model (see Sec. 2.7).

does not always know exactly with whom one wants to share which parts of the data being gathered[1028]. The workflow is as follows: The HMD senses the rate of the heartbeat and records it with a time stamp and a geographic location (geo.-loc.). When consulting the record, the medical personnel must be reassured that it was indeed taken by some HMD (e.g. certified medical equipment with a known good calibration) and not tampered (e.g. the patient faking to have done exercises). For protection, each triplet (timestamp, heartbeat, geographic location) is digitally signed on the HMD, using a secret signature generation key (sk) only known to that HMD. This is done for each new measurement; therefore the HMD not only signs but also *updates* data. Communication step 1 shows that signed data is sent to the patient's mobile phone. The patient's personal Android smart phone then plays the role of the privacy-enhancing gateway and selects which data is forwarded to the healthcare provider. For example, the timestamps are made more coarse and the geographic location can be removed in certain user-chosen areas. The patient entrusts his device and software to do these redactions. But the redaction itself requires no additional keys, the signature can be adjusted to fit the reduced data, e.g. geographic location information is removed in the workflow depicted in Fig. 137. The redacted and still verifiably signed data is sent in step 2 to the healthcare provider. The provider can verify the received reduced data's adjusted signature on a cloud-based back-end. In step 3, it sends the data to a healthcare personnel's tablet computer. Additionally, the workflow allows to further reduce the data set. Thus, by redacting sensitive and unneeded information from the data first, any service in the cloud can further tailor the still authentic information into smaller packages. This is depicted by the communication steps 4 and 5 in Fig. 137. Overall, there are three important advantages from facilitating an RSS in this way:

- The workflow requires no interaction with the HMD to tailor the data after the production into different packages for different healthcare usages or different personnel; still it allows any privacy-aware party to implement the need-to-know principle in an ad-hoc fashion.
- The workflow allows to verify that other than potential removal of more data, the data that is available has not been tampered with and originated from an identifiable HMD.
- The party or parties doing privacy preserving redactions are not required to be entrusted not to introduce other errors or even tamper the data.

Thus, the workflow achieves to keep a high data quality: The authenticity of the information shown to the healthcare personnel is protected even against a manipulation on the patient's smartphone, e.g., falsifying the heartbeat or geographic location to simulate jogging; and the origin is authenticated to be the certified HMD. This use-case has a small number of elements to be signed (<20) for each reading, but only *very* limited computing resources.

18.2.2. Implementation and evaluation on three platforms

The only known construction of a division-intractable hash-function includes expensive primality tests [127]. Henceforth, it is referred to as 'Prime'. With these tests the original $merge\mathcal{RSS}$ construction presented in Sec. 14.12 is cryptographically secure in the standard model[1029]. For the sake of reduced runtime the implementation of a second construction was devised which simply requires the hash's value to be odd and is henceforth denoted as 'Odd'. This reduced the cryptographic strength; it is provably secure[1030] only in the random oracle model (ROM). The security proof can be found in [198]. Still, the random oracle model allows to obtain a usable and for many workflows satisfying level of security. All hashes have been implemented as full-domain hashes using concatenation of shorter hash-outputs, as described by *Bellare and Rogaway* [37], paired with rejection sampling.

The crypto primitive was implemented in the C programming language for the constrained device RE-Mote and as a Java library for use in the cloud and on Android. Java was chosen because it is a widely used programming language and it provides many useful functionalities as well as existing cryptographic libraries. Furthermore, it allows re-using the same code base on both the Android and the cloud platform. The Android, Cloud and RE-Mote platforms are very different in terms of available computing power.

[1028] Not knowing what needs redaction later is also a problem when data is later prepared to be used in anonymous data; "Therefore, it is absolutely necessary to anonymize medical data by removing patient identifying information or by providing solely parts of the documents which are relevant for a specific study. But this can only be accomplished on demand and depends on the context. This means, that a priori anonymization of medical data is not feasible without knowing the intention of the study and the respective sample." [444].

[1029] This is also the version of the Construction published as joint work in [387] (see Appendix A publication n° 20).

[1030] As defined in Sec. 2.3.1.

A *cloud computer* uses multicore processors and has multiple gigabytes of random access memory (RAM). In the implementation of the healthcare use-case, it was run on an *Intel* Core i7-4790 quad-core processor clocked at 3.60 GHz to 4.00 GHz with 16 GB of RAM and Oracle's JVM in version 1.8.0.92.

A Motorola Moto G (3rd Gen.) smart phone was used as the *Android device* for the implementation. It has a Qualcomm Snapdragon 410 quad-core processor clocked at 1.4 GHz with 1 GB RAM and was running Android Version 6.0.

The RE-Mote is a *resource-constrained IoT platform* that has been designed for applications with additional cryptographic support for increased security and privacy [396, 473] and still offers a low power consumption. It is based on a CC2538 ARM Cortex-M3, with up to 512 KB of programmable flash and 32 KB of RAM.

18.2.2.1. $\mathit{merge}\mathcal{RSS}$ key generation is done in the cloud

The generation of keys in the use-case solely happens in the cloud (see Tab. 34 for performance). Still, providing a complete PKI was out of scope of this prototype's implementation. Nevertheless, the Java library needs access to the keys. This problem is tackled by loading the keys from a Java keystore, which is based on the keystore provided by the Bouncy Castle [310] crypto library. In the health-care scenario, having the keys generated in the cloud is helpful due to the high need for computation power: The search for large safe-primes on the constrained device could easily take a couple of days and it is foreseen that these keys are to be provided or indeed generated during a burn-in phase of the certified HMD.

18.2.2.2. $\mathit{merge}\mathcal{RSS}$ in the cloud

To use the computing power of the cloud environment to full advantage, the implementation of the sign, update and verify algorithms takes advantages of parallelisation. Namely, these three algorithms perform modular arithmetics and compute cryptographic hashes. This is more computationally intensive than removing elements from a given set, which is what the merge and redact algorithms do. To no surprise the merge and redact operations are very fast compared to the sign, update and verify operations. Thus, no effort into parallelising merge and redact was spent. Sign, update and verify perform computations on every element of a given set or message. This calculation is on one element only and it is independent[1031] from the calculations on all other elements of the message. Using this knowledge allowed to parallelise sign, update and verify; using Java methods for parallelisation the algorithms carried out the computation on each element of the set in parallel.

18.2.2.3. $\mathit{merge}\mathcal{RSS}$ on a mobile phone

The focus of interest is to measure performance. Hence, no user interaction was foreseen for the proof of concept implementation of the RSS on Android. The bare-bone Android application does not have a graphical user interface. The Android application reused the Java RSS library already developed and implemented for the cloud. All cryptographic algorithms and the testing benchmarks used for the cloud platform were also run on Android, except those concerning key generation. Key generation tests were excluded since in in the proposed scenario only the public keys are needed for verification. The quad core CPU of the Motorola Moto G allows to use parallel processing using features provided by the Java RSS library. This sped up the tests significantly but diminished the response of other applications and thus if done like this in practice it would negatively influence the responsiveness and thus the usability of the mobile phone. No further attempt for optimisation in that regard was made, as this was considered as not important for the bare performance tests.

18.2.2.4. $\mathit{merge}\mathcal{RSS}$ on a constrained device

The sign and verify algorithms are implemented in the C programming language, compatible with the Contiki operating system ran by the RE-Mote device. Functions from an existing Elliptic Curve Cryptography library[1032] for arithmetical operations on big integers were used. To tackle the considerable amount of time required by the safe prime generation algorithm, again the keys got generated externally

[1031] does not need a result of a previous calculation

[1032] `https://github.com/oriolpinol/contiki` [last accessed: Aug. 2017] and got simply hard coded on the RE-Mote device for the speed test. The implementation on the RE-Mote is not as fast as initially hoped. The possibility of improving the code by using a better optimised function for modular exponentiation, since this is the most expensive calculation in *merge*$\mathcal{RSS}$ and also called very often, might be a promising route to a more optimised implementation on a RE-Mote device. Another limitation of the RE-Mote is its small[1033] RAM size which made it impossible to sign or verify large messages.

18.2.3. Evaluation and observations

For every test the different parameters like the security parameter 1^λ and different levels of parallelisation as well as using the 'Prime' and 'Odd' hash-function implementation was varied. On the constrained device, due to the memory limitation, a message with only 4 redactable elements was used. For the mobile phone and the cloud, the tests were carried out using a message with 100 elements for the 'Prime' version and for those using the 'Odd' version of the hash-function, messages with 100 and $1,000$ elements were measured. Tests which took longer than 12 hours ($> 0.5d$) were stopped as such a runtime makes the scheme impractical to use. The test with $1,000$ elements using the original — 'Prime' — version of the hash-function was not attempted as it was expected to be too slow.

Most important of all, the measurements confirmed the expected speedup due to the revised hash-function, which does no longer require primality tests. It is denoted as 'Odd'. This resulted in an average speedup of $1,100$ times compared to the originally proposed one, denoted 'Prime'. Fig. 138 shows this speedup as well as how the construction of *merge*$\mathcal{RSS}$ scales well on multi-core CPUs.

Further, the test results show that — as expected — the runtime of operations of *merge*$\mathcal{RSS}$ increases linearly with the number of elements in a message. Additionally, it can be noted that the redact and merge operations are extremely fast independent of the used security parameter. Regarding the update and sign operation, it was expected that update would be slightly faster, since sign needs to additionally process the tag. However, this did not show in the results. Also the linearity of the selected RSS shows its practicality regarding update: updating n elements to a small message is equally fast as updating the same amount to a large message.

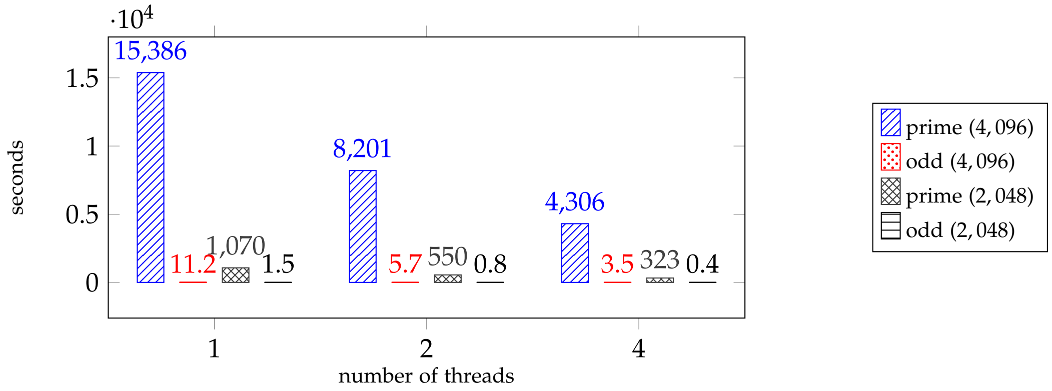


Figure 138. *merge*$\mathcal{RSS}$ Performance: runtime in seconds of signing a 100-element message with $4,096$ and $2,048$ bit keys on an Intel Core i7-4790 CPU

	Cloud		**Mobile**		**RE-Mote**	
	2,048	**4,096**	**2,048**	**4,096**	**2,048**	**4,096**
KeyGen	122.4	4,336.0	–	–	–	–
Redact[1034]	0.27×10^{-3}	0.19×10^{-3}	5.87×10^{-3}	4.17×10^{-3}	insuff. RAM	insuff. RAM
Merge[1035]	0.08×10^{-3}	0.06×10^{-3}	1.32×10^{-3}	2.813×10^{-3}	insuff. RAM	insuff. RAM

Table 34. *merge*$\mathcal{RSS}$ Performance: runtime in seconds of the KeyGen, Redact, Merge operations; as jointly published in *Frädrich, Pöhls, Popp, Rakotondravony, and Samelin* [198]

[1033] Small comparable to computers and smart phones, but the RE-Mote's RAM of 32 KB is already quite big compared to even more constraint IoT devices.

[1034] Redact 500 elements of $1,000$-element message.

[1035] Merge two 500-element messages.

merge$\mathcal{RSS}$ Sign		**Threads/1$^\lambda$**							
		1		2		4		8	
		2,048	**4,096**	**2,048**	**4,096**	**2,048**	**4,096**	**2,048**	**4,096**
Cloud	Prime (100)	1,070.0	15,386.0	550.0	8,201.0	323.0	4,306.0	294.0	4,161.0
	Odd (100)	1.5	11.2	0.8	5.7	0.4	3.5	0.4	3.0
	Odd (1,000)	15.0	111.0	8.1	56.3	4.5	31.3	3.9	28.8
Mobile	Prime (100)	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	–	–
	Odd (100)	13.4	93.5	6.7	47.6	3.7	25.1	–	–
	Odd (1,000)	132.0	925.0	65.9	465.0	34.3	254.0	–	–
RE-Mote	Prime (4)	$> 0.5d$	$> 0.5d$	–	–	–	–	–	–
	Odd (4)	962.0	$> 0.5d$	–	–	–	–	–	–

Table 35. *merge*$\mathcal{RSS}$ Performance: average runtime in seconds of signing 100- and 1,000-element messages; as jointly published in *Frädrich, Pöhls, Popp, Rakotondravony, and Samelin* [198]

merge$\mathcal{RSS}$ Update		**Threads/1$^\lambda$**							
		1		2		4		8	
		2,048	**4,096**	**2,048**	**4,096**	**2,048**	**4,096**	**2,048**	**4,096**
Cloud	Prime (100)	1,022.0	15,996.0	526.0	8,157.0	310.0	4,548.0	270.0	4,327.0
	Odd (100)	1.5	11.1	0.8	5.6	0.5	3.0	0.4	2.9
	Odd (1,000)	15.0	111.0	8.1	56.1	4.5	31.5	3.9	28.7
Mobile	Prime (100)	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	–	–
	Odd (100)	13.2	92.6	6.6	46.5	4.0	24.7	–	–
	Odd (1,000)	132.0	925.0	65.7	462.0	38.3	279.0	–	–

Table 36. *merge*$\mathcal{RSS}$ Performance: average runtime in seconds of updating 100 and 1,000 elements to a message; as jointly published in *Frädrich, Pöhls, Popp, Rakotondravony, and Samelin* [198]

merge$\mathcal{RSS}$ Verify		**Threads/1$^\lambda$**							
		1		2		4		8	
		2,048	**4,096**	**2,048**	**4,096**	**2,048**	**4,096**	**2,048**	**4,096**
Cloud	Prime (100)	1,047.0	18,911.0	553.0	9,724.0	334.0	5,416.0	300.0	5,080.0
	Odd (100)	1.5	11.1	0.9	5.6	0.5	3.2	0.4	3.0
	Odd (1,000)	14.7	110.0	7.4	55.3	4.8	30.4	3.8	28.6
Mobile	Prime (100)	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	$> 0.5d$	–	–
	Odd (100)	13.0	92.9	6.6	47.2	3.7	24.8	–	–
	Odd (1,000)	130.0	919.0	64.8	461.0	35.7	269.0	–	–
RE-Mote	Prime (4)	$> 0.5d$	$> 0.5d$	–	–	–	–	–	–
	Odd (4)	951.0	$> 0.5d$	–	–	–	–	–	–

Table 37. *merge*$\mathcal{RSS}$ Performance: average runtime in seconds of verifying 100- and 1,000-element messages; as jointly published in *Frädrich, Pöhls, Popp, Rakotondravony, and Samelin* [198]

The Construction 10 implementing *merge*$\mathcal{RSS}$ (see Sec. 14.12.2 on page 412) checks if the given message and signature verifies[1036] before calling the actual merge, update and redact routines. However, for pure performance, signature verification in these algorithms was disabled to gain a better impression of how fast or slow these operations are, because otherwise they would have included the verification runtime. For a full runtime of update, merge and redact one must add the runtime of the verification algorithm on top.

Tables 34-37 show the detailed results of the performance measurements; they give an indication of the average runtime of four repetitions of every test.

18.3. Concept: Information source authentication and integrity preserving application of privacy-enhancing technology (PET) in the smart energy grid

This section describes the positive results that can be obtained when RSS are applied to energy consumption data in a smart grid scenario. Those have been published in the joint work with *M. Karwe* at SMARTGRIDSEC 2014 [383] (see Appendix A publication n^{o} 18).

[1036] The algorithms state this step explicitly, e.g. for Redacting the algorithmic description states "1. Check the validity of σ using Verify. [...]".

18.3.1. Using an RSS to control privacy-enhancing integrity violations of energy meter output

As the technology around smarter energy grids is currently developing, fundamental security requirements like integrity and origin authentication need to be addressed while minimising arising privacy issues. The problem is to balance data utility (including integrity and authenticity) and data privacy. On the one side, the smart grid (SG) stakeholder needs access to integrity-protected values gathered by a trusted untampered smart meter (SM). On the other side, the consumer requires some trusted privacy component to perform data perturbation to protect the consumer's privacy. The main point to raise is that the entity trusted to generate data is controlled and trusted by the SG stakeholder. With its goals and incentives to gather fine-grained data, this entity is untrusted to maintain the consumer's privacy. Vice versa, the SG stakeholder will not be able to rely on data gathered by an untrusted consumer-controlled device. Fig. 139 depicts this situation.

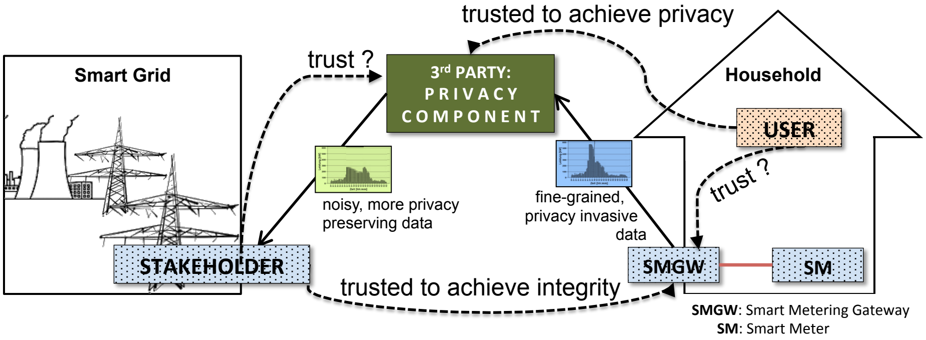

trusted to achieve privacy
trust ?
Smart Grid
3rd PARTY: PRIVACY COMPONENT
Household
USER
trust ?
noisy, more privacy preserving data
fine-grained, privacy invasive data
STAKEHOLDER
SMGW
SM
trusted to achieve integrity
SMGW: Smart Metering Gateway
SM: Smart Meter

Figure 139. Trust towards components by SG stakeholders and privacy-aware household; as jointly published in *Pöhls and Karwe* [383]

Secure RSS can help balance these two opposing goals. On the one hand, RSS will allow to change the data in accordance with privacy-enhancing techniques (PET) to mitigate privacy issues raised by overly precise energy consumption values via data perturbation mechanisms, e.g., add noise. On the other hand, the RSS can be instrumented to limit the noise's range and keep a verifiable level of integrity of consumption values from the Smart Metering Gateway by using a redactable signature.

The example shows the applicability of a private RSS that enables to delegate the content removal for increased data protection to a partly trusted party while still retaining a legally valuable signature. This showcases the need for achieving both: The cryptographic property of privacy for data protection and retaining a high probative value for the data originating from a trusted signer.

Several works show that highly fine-grained energy values allow detecting appliances within the household [351], detecting the use mode of the appliances [164] as well as deducting the residential customers' behaviour [317]. To mitigate those threats, current research and governmental organisations suggest using Privacy Enhancing Technologies (PET), e.g., albeit reducing the data quality. The proof of concept shown here indicates that such a reduction in data quality by redaction using an RSS is actually possible.

A reduction in data quality is also essential for increasing privacy, because even with lower data quality, simplistic algorithms can detect behavioural patterns like 'at home' versus 'away' from energy consumption readings. This result was published in joint work done with *B. Petschkuhn* and *J. Rückert* and *M. Mössinger* [397] published at IEEE CAMAD 2014 (see Appendix A publication n^{o} 40).

The remainder of this section describes how to use RSS to construct a technology that allows balancing the conflicting interests of privacy and integrity[1037]. For the PET it follows an approach called data perturbation, which is widely used in the field of privacy preserving data mining and differential privacy [159]. Data perturbation based mechanisms preserve privacy of distinct customers by letting an entity tamper with the data. For the smart grid this entity will be called the *privacy gateway* (PGW).

[1037] Integrity in this usage includes the protection of accuracy.

The downsides of data perturbation are twofold: First it obviously must result in a reduced data utility and second the data tampering entity must be trusted. The first is an inherent problem of PET whereas the impact on utility needs to be limited to a level where the application is still executable. RSS counter the latter by applying a redactable signature instead of a classical digital signature at the SMGW.

Note, the concept is built in accordance with a framework from the German Federal Office for Information Security (BSI) depicted in Fig. 140. In a nutshell the concept allows the amount of integrity violations needed to achieve the privacy to be controlled by the smart grid stakeholder. The smart grid stakeholder controls it by letting the SMGW sign a controlled *range of values* around actual energy consumption using a redactable signature scheme (RSS). The residential customer's *privacy gateway* (PGW) still has the possibility to 'tamper' with the data in order to increase privacy by choosing *one value* out of the signed range.

The redactable signature that is generated allows the Verifier to gain reassurance that the smart metering gateway (SMGW) actually signed this value. Hence, the Signer limits allowed values according to maximum tolerable reduction of data utility. For the privacy of the consumer the redactable signature allows the customer (or third parties) to do the choosing without any interaction with the Signer. Hence the customer does not need to trust a third party like a Smart Metering Operator (SMO) or the Smart Metering Gateway Administrator to protect the customer's privacy.

18.3.2. System description and integrity requirements

The German Federal Office for Information Security (BSI) proposed a technical guideline [73] for intelligent metering systems. While this technical guideline is controversial discussed in literature due to its broad as well as expensive security and its slim privacy concept [495], it allows for a controlled data communication between a household and smart grid stakeholders. The SMGW is the central point for the communication between the data requesting stakeholders (DSO or DSM) and the smart meters inside the household. Stakeholders can ask the SMGW to get consumption data. The SMGW can check if they are authorized. The time interval for the gathered consumption data may vary, for example in the EU a collection rate of '15 minutes averages' was discussed [185] and considered to be sufficient to guarantee net stability.

18.3.3. Privacy threats in the smart grid domain

Service providers in the smart grid like DSO or DSM need to collect data from individual households for their services. This data allows to infer information about households. The general research focus for privacy incursion has been about energy consumption values which are considered the household's output channel. Note that research barely considers the other direction, the input channel to the household. Inferred information of energy consumption values can be structured in the following three categories:

1. appliance detection,
2. use mode detection[1038], and
3. behaviour detection.

Note that all these attacks are assumed to be possible for any party that has access to the plain energy consumption data. Hence, encryption will help to protect the confidentiality during transmission of data, i.e., to achieve privacy against third-parties, but will not mitigate privacy attacks by the party that legitimately receives it and can thus decrypt it to get the plain data.

In the first category an analyser tries to find out which appliances run in a household site. This information can be used for advertising purposes. In the second category an analyser tries to find out how those devices are used. Experiments with high frequency data shows that even the TV channel can be deduced with a high percentage rate [218]. In the third category data is used to investigate how many people live in a household and what those people do. In *Lisovich et al.* [317], wake and sleep cycles as well as presence and absence have been successfully deduced.

[1038] Following *Jawurek* use mode detection"[...] tries to infer the actual activity performed with the device." [272].

The information transmitted over the channel from SG service providers to the household bears privacy risks which depend on the application. Demand Response (DR) application allow to infer incentive sensitivity as well as a customer's preferences. In a simple version of a demand response application, the DSM asks the customer to reduce the amount of consumed energy in a certain time frame. In return the customer gets a financial compensation. To measure the compensation amount the DSM needs to know the energy consumption of this time as well as data to compare in order to determine the real reduction. This data can be the consumption from former periods. With this data and to know when the customer accepts and executes DR requests, the DSM can infer incentive sensitivity information of the customer.

18.3.4. Perturbation as a privacy-enhancing technology

To mitigate the privacy threats of appliance- and use mode detection as well as behaviour deduction, several privacy-enhancing technologies have been introduced. Privacy enhancing technologies (PET) can be based upon the principle of data minimisation and concealing. The main drawback of those techniques is that either customers can not be addressed individually or that fine granular data is not available. A different approach than data minimisation and concealing is the addition of noise to consumption data. While the outlook from the standpoint of privacy protection is very promising, the effect of the introduced error to data utility in the smart grid is still in research. In general, data perturbation, done in a right way, has been shown to reach the differential privacy ad omnia guarantee [159].

Note, the joint paper [383] discusses differential privacy in more details, however, they are not relevant for the understanding of the RSS application and are thus omitted (see Appendix A publication n^{o} 18).

18.3.5. RSS with high probative value to control the PET's invasiveness and the resulting data quality

When energy consumption data is signed using an RSS with a high probative value the SMGW provides the smart grid stakeholders like DSO and DSM with signed and henceforth more trustable smart meter values, e.g., energy consumption values. At the same time, it allows the customer to achieve a desired level of privacy by allowing the authentic energy consumption value to be tampered with, e.g., by adding noise. The party running the PETs to achieve the consumer's privacy is termed Privacy Gateway (PGW). The proposed solution is depicted in Fig. 140. It is assumed that all information between the smart meter gateway (SMGW) and the DSO or DSM a are running over the privacy gateway (PGW) as the curator. Hence, the terminology of calling PGW a 'gateway'.

Note that it is the smart grid stakeholder who knows and requests a desired level of data utility. In case of a perturbation by noise this requires to limit the maximum allowed noise. The maximum noise in each direction is denoted as δ_{max}. Of course, the SMGW could run privacy preserving algorithms directly and add noise to keep the customer's differential privacy. However, this solution would require that the residential customer trusts the smart meter operator (SMO) to protect the customer's privacy. The same problem occurs vice-versa if the PGW is placed before the SMGW, but would directly tamper with the readings from the smart meter. Then the DSM and DSO have to fully trust the PGW. However, the proposed solution allows the party doing the addition of noise just to be trusted to preserve the customer's privacy, as the customer remains in full control of what is selected, but the PGW can not create readings not initially generated by the SMGW. In short, the task of the newly proposed PGW is to tamper energy consumption values in order to protect the privacy of residential customers. The task of the already existing SMGW is to sign the energy consumption values and additionally the maximum tolerable perturbation in order to protect the integrity and trustworthiness of the smart meter (SM) readings. Both parties act on behalf of different stakeholders and hence are in different trust zone. With the suggested solution of applying a redactable signature schemes (RSS) this conflict is solved.

Construction 15 : RSS to control perturbation

> *The proposed solution is to apply an RSS to control the perturbation, i.e. adding noise, and keep the origin's authenticity for the remaining data in tact. It consists of four algorithms for four phases: Setup, signing, adding noise and verification.*

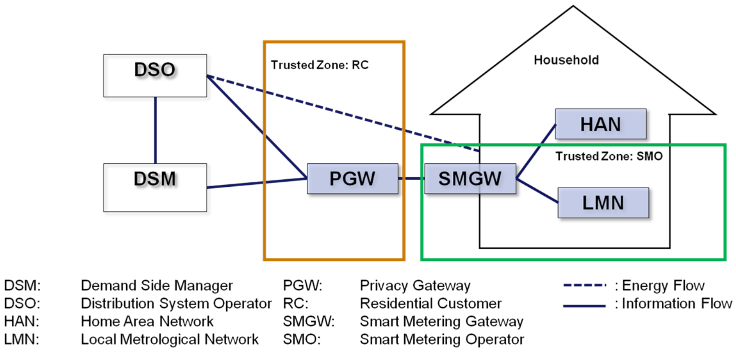


Figure 140. System structure with the privacy gateway (PGW) trusted by the residential customer (RC) in front of smart metering gateway (SMGW) trusted by the smart metering operator (SMO); as jointly published in *Pöhls and Karwe* [383]

Setup:

1. *Let* RSS $:=$ (KeyGen, Sign, Verify, Redact) *be a secure (unforgeable and weakly private) redactable signature scheme.*
2. *After running* KeyGen *distribute the keys: SMGW gets a secret signing key* sk *and verification key* pk*, PGW and DSO/DSM get just the public SMGW's verification key* pk*.*
3. *SMGW is instructed by SMO which amount of noise it tolerates, or alternatively which accuracy is required.*

Signing:

1. *On receiving the actual consumption value v the SMGW calculates a range of discrete noisy values $\mathcal{M} = \{v - \delta_{max}, \ldots, v, \ldots, v + \delta_{max}\}$.*
2. *SGM signs $\mathcal{M}$ with an* RSS*: $(\mathcal{M}, \sigma) \leftarrow \mathsf{Sign}(1^\lambda, \mathsf{sk}, \mathcal{M})$.*
3. *SMGW sends $(\mathcal{M}, \sigma)$ to PGW.*

Adding Noise:

1. *On receiving $(\mathcal{M}, \sigma)$ PGW uses its database of historic values and the actual consumption value, which must be at the centre of the range in $\mathcal{M}$, PGW runs the differential privacy algorithms to identify the value n in $\mathcal{M}$ which should be sent to DSO/DSM in order to satisfy $\frac{Pr(k(D_1) \in S)}{Pr(k(D_2) \in S)} \leq e^\epsilon$ where ϵ is a user predefined minimum required privacy parameter. The application execution is denied, if ϵ can not be reached.*
2. *PGW calculates $\mathcal{R} = \mathcal{M} \setminus n$.*
3. *PGW obtains a signature on $\mathcal{M}' = n$: $(\mathcal{M}', \sigma') \leftarrow \mathsf{Redact}(1^\lambda, \mathsf{pk}, \mathcal{M}, \sigma, \mathcal{R})$.*
4. *PGW sends $(\{n\}, \sigma')$ to the DSO/DSM.*

Verification:

1. *On receiving $(\{n\}, \sigma')$, DSO/DSM uses the SMGW's verification key* pk *to verify if the signature on n is valid.*

The amount of elements in $\mathcal{M}$ depends on the maximum noise and the accuracy, as $\mathcal{M}$ must contain concrete values, e.g., $\mathcal{M} = \{0.99, 1.00, 1.01, 1.02, 1.03, \ldots, 1.48, 1.49, 1.50, \ldots, 1.96, 1.97, 1.98, 1.99\}$ for an accuracy of two decimals, $\delta_{max} = 0.50$ and $v = 1.49$. The RSS limits the PGW only to redactions based on provided values, e.g., for $\mathcal{M} = \{1.11\}$. The PGW could generate a valid signature facilitating the algorithm Redact. However, the PGW can not generate valid signatures on values outside the range, e.g., $\mathcal{M} = \{0.98\}$ or $\mathcal{M} = \{2.00\}$. To do so would be as hard as forging the signature scheme of the RSS, e.g., breaking the signature scheme like RSA-PSS [35, 408]. To counter replaying or repressing messages, the SMGW can just add a timestamp as an additional element into $\mathcal{M}$ requiring this to be fresh and present during verification.

18.3.6. Security and privacy properties gained with the RSS

The concept assumes the following: The smart meter (SM) is trusted to perform correct readings, can not be attacked, and transmits the reading securely to the smart meter gateway (SMGW), this forwards the data to the privacy gateway (PGW).

Theorem

The protocol is unforgeable, i.e. an attacker can not generate a valid signature for a value n^* that is not in the range signed, if the RSS is unforgeable.

Proof (Intuition)

If the RSS applied by the SMGW is unforgeable, then neither PGW nor attackers can forge a valid signature on a value $n^* \notin \mathcal{M}_i$, where $\mathcal{M}_i$ denotes all sets signed and sent by the SMGW. Any such forgery would be a forgery in the RSS. □

Theorem

The protocol allows to achieve the highest differential privacy possible for $\Delta = 2\delta_{max}$ while keeping authenticity of origin, if the RSS is at least weakly private.

Proof (Intuition)

Assume all communication from SMGW will always pass through the privacy gateway (PGW), see Fig. 140. The RSS allows PGW to be a separate entity acting as instructed by the residential customer. PGW is limited by the range defined within the SMGW's signature but can run the algorithm Redact to select any suitable value out of the range. So seeing a valid $(\mathcal{M}, \sigma)$, which verifies using Verify under the trusted public verification key of a SMGW, that no malicious modification has taken place. Privacy of the underlying RSS guarantees that attackers can not identify the actual value of removed elements. Hence the adversary can not know the actual consumption. Here one can distinguish two cases:
(1) If the RSS is strongly private, i.e., elements are completely removed during redaction, then the adversary sees a set $\mathcal{M}$ with exactly one element, i.e., $|\mathcal{M}| = 1$.
(2) If RSS is weakly private, i.e., original values are hidden behind a special symbol ($\blacksquare^r$), then the adversary sees a set $\mathcal{M}$ with exactly one element being an actual value and $2\delta_{max}$ symbols, i.e., $|\mathcal{M}| = 2\delta_{max} + 1$.
Hence, if RSS is weakly private and the adversary knows the precision of the values, then an adversary can infer δ_{max}. However, the adversary does never learn the actual values of removed elements, nor their position because it is a set of values. Using the differential privacy mechanism described in the original joint work [383], the PGW selects the noise to add within the range guaranteeing a differential privacy of ϵ. □

For any application of smart metering it is vital that the SG stakeholders receive *reliable* and trustworthy information. In this case *reliable* means that the SG stakeholder, e.g., a power grid provider, gets this information as (1) timely and as (2) accurate as needed for the SG application. The exact level of accuracy and timeliness will vary depending on the application itself, but also on the actual contractual, regulatory and installation setting, and is beyond the scope of this concept. In the concept the SM operator (SMO) limits the range in which data perturbation, in this case the addition of noise, is considered acceptable by applying a redactable signature (RSS) at the SMGW over a range of the SMO's choosing. Knowing the allowed level of accuracy allows the customer's privacy gateway (PGW) to calculate the differential privacy guarantee that it could achieve using the data perturbation mechanisms it could deploy. With this information the PGW can independently judge if the allowed perturbation is enough to keep a sophisticated level of privacy for the customer. If not, it can withhold the information until the customer explicitly consents to this leaking of PII. If the PGW has enough freedom it will adjust the data accordingly and forward it after the modification. A RSS allows this alteration of signed data and the SG stakeholder can verify if the change was within his defined limits.

EXAMPLE

Example: Assume the following values $\{3.5, 3.6, 3.7, 3.8, 3.9\}$ being signed by the SGMW. The PGW knows that the original reading is 3.7 but decides to sent 3.5 to the DSM. Then PGW actually sends a set $\{\blacksquare, \blacksquare, \blacksquare, \blacksquare, 3.5\}$, but sending $\{\blacksquare, 3.5, \blacksquare, \blacksquare, \blacksquare\}$ would be as good because its a set. If the verifier knows that all values are of the same precision (0.1) then it can infer from the set's size being 5 that $\delta_{max} = 0.2$. Knowing that the real, un-tampered, reading of the SMGW was in the range of 3.3...3.7.

The presented idea differs slightly from the general idea of differential privacy. In differential privacy ϵ is chosen under the perspective to protect privacy. The concept's idea is to regard the application side and limit the noise to its needs. This allows calculating the ϵ depending on the maximum amount of noise that the data perturbation mechanism k is allowed to apply. The amount of noise is defined by the max. error which is acceptable for an SG application.

This approach can be criticised for its weak privacy protection. Very small noise will allow appliance detection and behaviour deduction. It remains unclear to which extent this small noise prohibits use mode detection. Due to the need of very fine grained data to get an acceptable quality level for use mode detection it is assumed that the reduced accuracy introduced by the added noise will limit invasiveness.

It can also be argued that as the SG stakeholder controls the amount of noise, it can limit the privacy protection by setting a too low limit. Further investigation and discussion for concise applications with known data quality needs is required.

18.3.7. Implications of RSS signatures with an increased probative value

The RSS needs to be unforgeable and weakly private to allow for a removal of information by the privacy gateway (PGW). Further, the RSS needs to work on sets, i.e., leak no information about the position of the block from the signature. Then, all smart grid (SG) stakeholders can detect any subsequent malicious manipulation of information while it is travelling through the network. Additionally they can use the SMGW's verification key to identify, due to unforgeability, the origin of noisy data. In this concept, all but one single element from the set signed with an RSS shall remain after appropriate processing. Thus, the Verifier already knows that redactions will have taken place and thus the property of cryptographic transparency is not required. The use of a scheme like ARSS would allow to even make the redact operation key dependent, but this could reduce the user's freedom in choosing which privacy gateway to use will change the SM's reading, e.g. which privacy organisation's service or local components will run the perturbation service on behalf of the residential consumer.

Using a scheme which offers a high probative value, like $pubacc\mathcal{RSS}$, would make all parties publicly accountable. This would prevent the SMO as well as the SMGW to maliciously carry out a redaction instead of the intended privacy gateway (PGW) and later claim it was from the PGW. This could also be achieved with an additional signature from a CDSS by the PGW, thus removing the drawback of fixating the PGW during signature generation by an ARSS.

18.4. Concept: RSS for product content documentation of food supply chains

The following is another example of an application of MSS in the food supply chain. Different to the previous described SSS (see Sec. 18.1) this application applies an RSS to the food supply chain[1039] to enable to verify the authenticity of signed documents even if trade secrets or personally identifying information have been removed. Thus, private RSS with a high probative value allow sharing integrity-protected supply chain documents with a verifiable origin, such as orders or lab-reports more freely among the partners of a supply chain after the redaction has removed sensitive data. This exchange creates a more transparent supply-chain among the partners while respecting each partner's individual confidentiality requirements (trade secrets / employee data protection regulations). Another application is to utilise the above to the full extent: Allowing consumers to technically identify all the ingredients of any product. This data in the hands of consumers increases confidence and allows identifying potentially risky products that are already in the hands of the consumers. Further, it could aid the consumer in making her / his buying decision, e.g., the 100 mile diet [446], or other personal preferences.

[1039] The following results were published in the joint work with *F. Höhne* at the Interdisciplinary Conference on Current Issues in IT Security of the Max Planck Institute for Foreign and International Criminal Law (MPICC) [382] (see Appendix A publication n° 9).

EXAMPLE

For an example assume that in a food supply chain one would retain verifiable signature of every ingredient of the final product signed by the principal where the ingredient originated. The example is taken from the joint publication with *F. Höhne* [382] (see Appendix A publication n^{o} 9). In the exemplified chain for 'Spätzle' production (Fig.141), a so termed "sticky signature" [382] protects the ingredients lists of input products. Even if there are only two ingredients the number of participating suppliers may become large. For an organic product all the ingredients must be organic, too. In the example the ingredients are: organic non-GMO[1040] wheat flour and organic chicken eggs. Hence, organic chicken-feed must be used for egg production. The noodle producer is the last in the chain. The producers of input for chicken-feed like corn, sunflower and vitamins are the beginning of the chain. Of course, the noodles themselves can later become again an input amongst many.

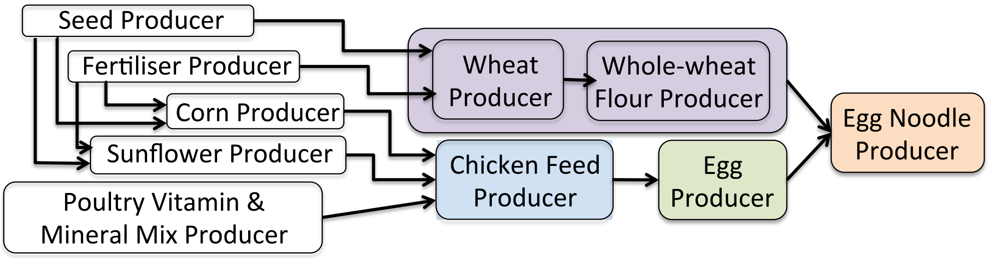


Figure 141. Simplified overview of supply participants in the production of egg noodles; as jointly published in *Pöhls and Höhne* [382]

This list is in itself again composed of and includes the previous stage's signed list in the output of the current stage, as shown in Fig. 142. The example assumes that supply chain participants can verify the credentials of all other participants[1041].

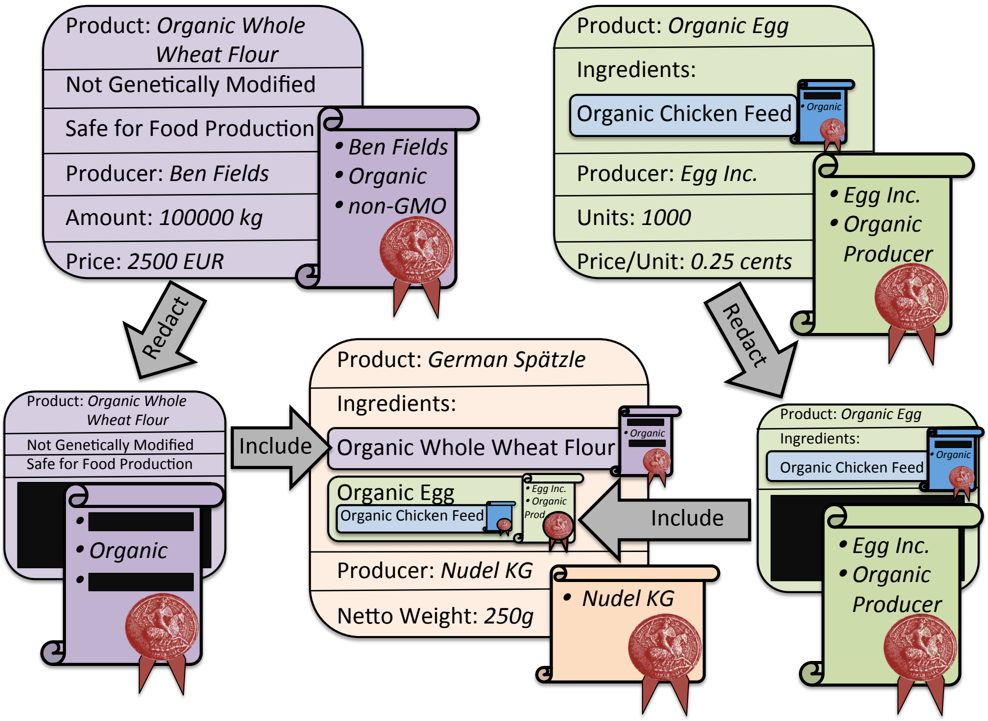


Figure 142. Sticky signatures: redacted signed data and credentials of inputs are included; as jointly published in *Pöhls and Höhne* [382]

[1040] not genetically modified chicken-feed

[1041] For simplicity, without loss of generality, the example further assumes a single trusted credential issuer (depicted as red seals).

Each producer, i.e., 'Ben Fields' gets a key pair for the RSS $(\mathsf{pk}_{Ben}, \mathsf{sk}_{Ben})$ and a credential C_{Ben} which contains among other information the public signature verification key pk_{Ben}. Each product is accompanied by important product information signed using the RSS with the producers sk_{Ben}. The document exchanged between the noodle producer *Nudel KG* and *Ben Fields* also contains information about the amount and the price of the 'Organic Whole Wheat Flour' that was delivered.
For simplicity every entity is allowed to redact every information from the product information that is deemed unnecessary at this step in the production. Still, third parties can verify the signature over the remaining information using the signer's public key and verify the proofs of attributes from the credentials to verify the origin of the signer. Using a private RSS means that once data is redacted using the RSS it cannot be reconstructed without knowledge of the removed submessage. Thus, 'Nudel KG' redacts unnecessary information before this data about the ingredient is included in the product information of the German egg noodles. All data necessary to judge the product's quality and to fulfil legal requirements remains non-redacted, e.g., all inputs are organic and from producers that are certified to be organic. For eggs they also state the egg's origin as required for egg products.

Note, as discussed in Sec. 8.3.5 the legal regulation for participants in food and feed production chains have to keep a legally transparent documentation of all their input products' suppliers (up-stream) and their products' purchasers (down-stream).[1042] Thus, each player in the supply chain archives the signed and non-redacted data as received from its supplier. When the authorities would demand to get more information than what is given in the redacted information that travelled alongside a food or feed product under investigation, then the original document can be handed out.

The above example requires that with the help of archived non-redacted versions a party is capable of undoing the blinding whenever the original authentic value is known and can be provided. Or at least the ability to prove that the archived non-redacted version is related to the redacted version. Technically this requires the non-redacted version to be linkable to a redacted version. Thus, this use case benefits from linkability. Note, this operation does neither violate the cryptographic privacy notion nor the legal one, as one is required to already know a previous version of the document. The example scenario especially motivates Requirement 8 (Mechanism MAY offer linkability of two signed documents if both have been created by subsequent, authorized modifications from the same signed source document (optional)) that was presented in Sec. 7.8.

18.5. Concept: Exercising data governance rights and give legal consent to flow of data

The following section integrates a secure and private RSS with an increased probative value to allow an automated processing of requests for personal information handled by a company. It is motivated by the problem that a process initiated at one company that was provided with a consumer's personal data usually involves more than one entity, e.g. a chain of companies. Additionally, each entity might not want to reveal too much information about their internals, i.e., neither about their predecessor (their client) nor about their successors (their subcontractors or suppliers). Therefore it needs to be standard private, e.g. allow to remove information that is considered a trade-secret. The signature on the requests and answers further serves the purpose to authenticate the originating entity in a legally binding way and hence an increased probative value is essential. The process is modelled as an orchestration following the paradigm of service-oriented architectures (SOA), e.g. following [326]. This kind of application scenario was also used to motivate the use of private RSS and SSS with a high probative value in several (joint) publications[1043].

[1042] Article 18 and 19 of EC Regulation 178/2002 [178] are requiring that "[f]ood and feed business operators shall have in place systems and procedures to identify the other businesses to which their products have been supplied. This information shall be made available to the competent authorities on demand." [178, Art. 18 para. 3].

[1043] The author of this thesis published the signed and redactable personal data tree at the ICICS conference in 2008 [375] (see Appendix A publication n° 4). A joint paper with *R. Herkenhöner, H. de Meer, and M. Jensen* is explaining the data governance rights requests and responses in service-oriented architectures presented at WSBPS 2010 [226] (see Appendix A publication n° 5). Finally, an RSS can also be used to indicate and preserve the consent for information processing when data is given to social networks as presented in the joint work with *A. Bilzhause, K. Samelin, and J. Posegga* [390] (see Appendix A publication n° 8).

According to legal obligations, every company that processes *personal data* must provide the *data subject* with means to query the company about what information is stored, processed, and exchanged with other companies. This *right of access* is usually provided as a manual process, requiring the data subject to place his request in person or via postal mail. Additionally, such a request will only cover the personal data regarding a certain company. In times of inter-organisational business processes and worldwide supply chains, this implies that the data subject has to query every company within such a business process in order to determine the full processing path his personal data took.

In contrast, a majority of inter-organisational business processes of today is realised using technical infrastructures like web services. It would be reasonable to provide information on processing of personal data according to the right of access as a dedicated web service itself. This way, the providing can be performed automatically, hence saving operational costs and enabling a company's customer to easily monitor the processing of his personal data. An SSS can help to build a web service to fulfil the legal requirements originating from data protection law in an automated way. However, a business process involving any data about customers will most likely result in a web service processing *personal data*. This is covered by data protection laws. In the following work, based on the joint publications, e.g. [226], the European data protection law codified in EU Directive 95/46/EC [174] was taken as the legal basis. The same is found in the updated Regulation (EU) 2016/679 [GDPR]. Hence, *personal data* "shall mean any information relating to an identified or identifiable natural person (*data subject*)" [Directive 95/46/EC] and [Art. 4 (1) GDPR]. In a common business process, the customer is the data subject and gives his personal data to a primary web service. The primary web service then forwards it, fully or partially, to its successors, and so forth. Each company that is given any part of that data — which is still considered personal — becomes a *controller*. A controller is required by law to grant the data subject the *right of access* among others. The right of access allows the data subject to determine which *categories of data* are being processed and the *purpose of the processing*. According to law, the controller's answer should also include the *sources* and the *recipients* of the personal data if forwarded to successors. In the following this process is called *providing of information*. However, this poses a problem for real-world business processes, since a company might consider business partners or customers a trade secret. To protect such trade secrets, a controller can reply with *categories of recipients* rather than identifying them directly. In complex business processes, such categorised answers lead the data subject's request to a dead end, as preceding and succeeding processing of personal data and involved controllers remains hidden. This is a major shortcoming in the existing practice of information providing.

18.5.1. Identified problems

A business process, realised as an inter-organisational service composition, does not automatically allow the data subject to exercise his right of access while preserving trade secrets of the participants. In current practice, requests of the data subject are handled by manual investigation and sending paper letters. This implies four major challenges to address:

- **Cost-efficient scalability**: A company taking part in a service composition has no efficient automated way to answer the data subject's requests. If business transactions increase, web services are able to scale, but manual processes involving human agents answering the requests do not. Companies have to find means to comply with legal requirements with minimal operational costs.
- **Authentication of requests**: An answer must only be given to a request from an authorized data subject.
- **Protecting trade secrets while answering requests**: The answer to a request on processing of personal data may violate trade secrets. Giving this information would allow to reconstruct all companies involved in a business process along with their position within it.
- **Confidential information providing**: The answer has to be delivered exclusively to the data subject, to prevent illegal disclosure of personal data.

Sanitizable signatures (SSS) provide a **solution** for these issues and can be used to extent existing service-oriented architectures (SOA) — based on web services technology — and allows implementing the right of access given by the European data protection law as a web service. The interested reader is directed to the published paper, but a small example of the concept is as follows.

18.5.2. Example of a data subject's request in a simple composed service

The user can send request for information in order to exercising his right of access. The data subject may address any of those business partners that are involved in a composed service. Fig. 143 depicts an example where the data subject was giving personal data to two companies C1 and C2. Those provided service by using a number of sub-service providers, e.g. imagine an order in an online store that involves shipping companies knowing your shipping address and banks doing credit checks. As such some personal data flows to C3–C7. As indicated the data subject is then sending its request to a company in order to exercise its right of access. The following assumes that the data subject is signing this request with an electronic signature in order to provide an authentication of the request. In the same way asymmetric keys can be used to encrypt the answer in order to provide the confidentiality of the response.

The initial request of the data subject is termed 'Right of Access:Request' or ROAR request in the joint work [226]. It is always directed towards a company within the service composition. The result of such a request will be a full report on that company's processing of personal data of the data subject. This report is called the ROAR response. Besides the ROAR request, a company may provide the additional service to ask its business partners about personal data of the data subject, insofar they were involved in the same business process. Hence, a company may request personal data of the data subject from its adjacent business partners, which will also be added to the ROAR response. This type of request is called 'Right of Access:Delegated' or ROAD and is only triggered by a ROAR request or a previous ROAD request. It is carried out between business partners that know and trust each other, and only if both companies are involved in a business process that previously processed personal data of the requesting data subject. As with the original business processes itself, the ROAR and ROAD requests and responses can be realised using web service technologies.

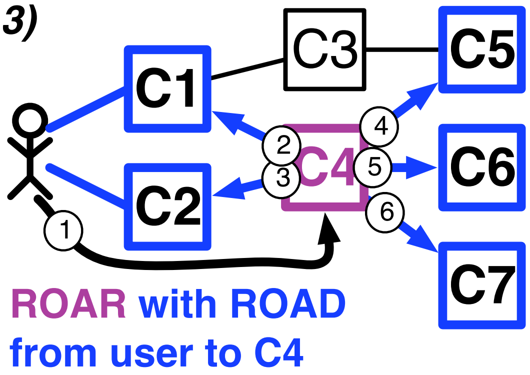


Figure 143. Flow of a request: The data subject directly sends a ROAR requests to C4, which results in further ROAD requests; only "Case 3" as jointly published in *Herkenhöner, de Meer, Jensen, and Pöhls* [226]

There are several scenarios for requests of the data subject discussed in the published joint work [226]. In the following the most straightforward example[1044] is depicted in Fig. 143. It illustrates a ROAR request from the data subject to a participant of the workflow, namely towards C4, depicted as (1) in Fig. 143. Assume the data subject is exercising its given right to ask: "Dear C4, tell me the personal data you hold from me? Tell me from whom you have received it? Tell me to whom you gave it to?" In order to answer this request C4 generates a number of ROAD requests (2–6) and those get propagated in both directions: towards predecessors and successors of C4 . Note, there are no ROAD requests sent to C3, as it is not part of the business processes involving C4, but was just used in a part of the process that involved C1. All the responses are collected and put in the ROAR response of C4 (see Fig. 144).

In Fig. 144 the structure of a ROAR response message from the example ROAR request towards C4 is depicted. It contains three distinctive blocks: Data sources, data processing information, and data recipients. For each ROAR response, a set of ROAD source blocks is given (disclosing the personal data's origins, here C1 and C2), as well as a set of ROAD recipient blocks of companies that directly received personal data from C4 (here C5, C6, C7). The third block contains the unique request identifier of the overall ROAR request as well as all information regarding C4's operations on the personal data. It lists

[1044] Originally this was case 3 out of four examples in [226].

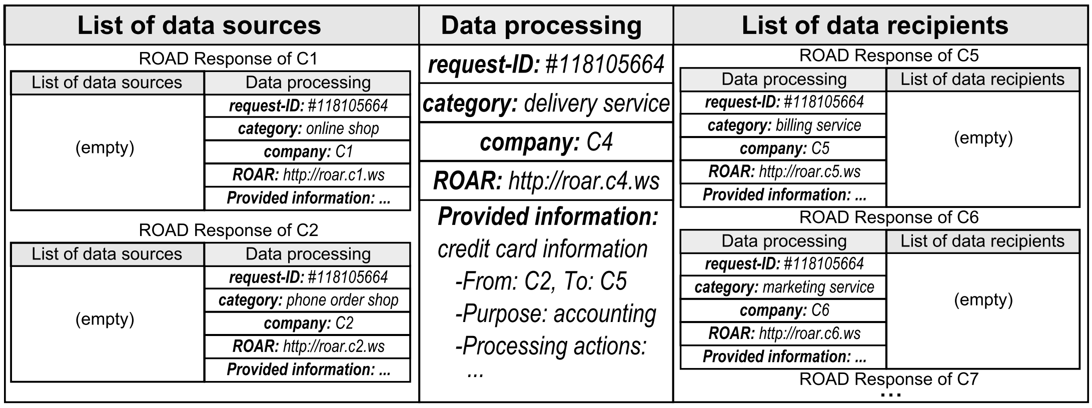


Figure 144. ROAR response message for a ROAR request to C4 of the case shown in Fig. 143; answer from C7 is compressed to [...] in the above List of data recipients for brevity; as jointly published in *Herkenhöner, de Meer, Jensen, and Pöhls* [226]

the company's business category (e.g., delivery), full name and a legal representatives' identity (e.g., Federal Express Corporation Inc. and Frederick W. Smith), ROAR service endpoint (in case the data subject wants to place another ROAR request), and a description of all operations done to the personal data of the data subject within the company. Within the last field, every processing step of personal data within a company is listed, denoting that data's origin, processing steps, purposes of processing, and recipients. The actual contents of this depend on the exact legal obligations of the data processor.

The included ROAD information of other companies contained within the ROAR response illustrate the result of the subsequent ROAD requests triggered by the ROAR request (here to C1, C2, C5, C6, C7). They contain the same kind of information as for the ROAR response, but list either additional sources (for data origin ROADs) or additional recipients (for data forwarding ROADs). Each such ROAD block also contains the same set of data processing information as the ROAR response, i.e., category, company, and provided information. For the special case of the data source being the data subject itself (e.g., for C1 and C2), the source list is left empty, and the data subject is denoted directly within the provided information field. The same applies for personal data being forwarded to the data subject, if ever.

Being a recursive message format, it is possible to illustrate the full spanning tree of data processing instances for the personal data. It discloses all companies involved in processing such data, along with their correlation. Hence, it gives the data subject a very broad view on the data processing network as a whole [1045].

18.5.3. Application of an SSS to guarantee authenticity of the response data while allowing to adapt responses to keep trade-secrets

Following the current German BDSG based on the European Union's — now soon to be updated — Directive 95/46/EC [174] the data subject has the right to access. This access rights is still in place in the new Regulation (EU) 2016/679 [184]. The laws intention is to guarantee any person the right of access to obtain information about the processing of his personal data.

However, there are limits. Those limits balance the right of access with the right to trade secret protection as follows: The right has to be exercised such that it "[...] adversely affect[s] [...] trade secrets [...]"[1046] [Recital 63 GDPR] but also that the result of the consideration of protection of trade secrets "[...] should not be a refusal to provide all information to the data subject."[1047] [Recital 63 GDPR]. To enable this protection of data sources and data recipients as a trade secret the law offers only one possibility to protect: To **categorise** the data recipient into a class instead o providing the actual name

[1045] Note that though it is possible for the chain of data processing instances to contain loops (i.e., a data item is sent back to a previous processing instance), this does not affect the proposed architecture, since the business process endpoint — and hence the processing logic — is different from the processing logic in the first pass. Hence, the ROAD response might contain the very same company more than once, but there is no threat of "infinite recursion".

[1046] About the same statement was also in the former Directive: [174, Recital 41].

[1047] A slightly modified wording was used in the former Directive as it stated that a consideration of rights to withhold information shall not result "[...] in refusing all information [...]" [174, Recital 41].

of the business entity in the answer. Namely, it is stated that "the recipients or categories of recipient to whom the personal data have been or will be disclosed [...]" [Art. 15 1 (c) GDPR] can be provided. As a result of providing only "categories of recipient" [GDPR], the data subject cannot gain knowledge on the entity that was given personal data for data processing beyond the point of categorisation. This means that the data subject is not able to direct request to that entity and thus would gain less information in regards to the given right of access. Especially, this usually results in no information about any further sub-processing is hold back from the data subject.

To resolve this conflict, the proposed solution based on a private SSS with increased probative value first delegates the request throughout the whole business process to collect information. The delegation is useful as every controller knows its predecessor and successor. Thus the request can be forwarded nevertheless a participant wants source or recipient to become categorised in the final answer to the data subject. It is left upon the controller (or upon her predecessors or successors), if it categorises its sources or recipients. The decision depends wether or not the controller considers them a trade secret. Thus, the controller must be able to sanitize signed answers it got from predecessors before including them in its own reply message. In a nutshell, the requesting entity will validate the sanitizable signature scheme on the responses and will see that either the original data or a blinded version was included by the responding entity. For ease of presentation, the SSS is configured such it allows everyone to blind data items, e.g. all participating entities get the sanitization key.[1048] Hence, the SSS allows for categorisation and enables trade secret protection while still providing a representation of the personal data's flow for the whole business process for which the authenticity can be verified by each requestor. Several other technicalities are required, but for brevity the discussion is not reproduced here, please see the original work [226] if interested. On prerequisite is that the data subject is required to digitally sign the personal data once before it gives it to any company as part of a business process execution. This can be done in a way close to what was explained in *Pöhls* [375] published at ICICS 2008 (see Appendix A publication n^{o} 4). That signature over a tree of the data subject's personal data serves as a data-subject-authenticator. It must be retained throughout all business processes along with the personal data. However, as personal data items may be split during a business process execution, this approach requires the use of sanitizable signature schemes in order for each controller to remove unnecessary data 'fragments' before forwarding while retaining the signature's validity which vouches for the personal data's integrity towards the next service in the composition. Note, categorisation is only effective, if all descendants or ancestors of an entity also categorise their answers. The proposed solution is to set an anonymity 'flag' in the request which is obeyed to by the upstream/downstream entities. It indicates that all further triggered ROAD requests are to use categorisation too. This solution assumes that disobedience of the flag violates the business contract with the predecessors or successors. Forensic investigation would reveal the violation, since the answer is signed by the controller, and allows imposition of liquidated damages.

The whole answer composed from such a delegated set of requests among the process partners reveals to the data subject the impact of processing the personal data in that specific business process. This would match the intended information of the data subject much better than having to send direct dedicated ROAR requests for every participant in an unknown business process, also it would miss information after the first categorisation.

EXAMPLE

Example: Assume a bank participating in a process and as a sub-process use a company for credit scoring the data subject might know that the shop used a service of the category 'bank', but it can not sent a direct request to *a bank* and thus would not know about that the data subject's personal data also was flowing from *a bank* to *a credit scoring service*.

The example highlights that the solution of delegated ROAD request allows to traverse the full business process. Thus it creates an added value, as it reveals the personal data's flow regarding the actual business process. Still, the proposed architecture offers a trade-off between preserving trade secrets and providing information to the data subject by facilitating the SSS to allow subsequent editing to change the exact entities' names into their categories. This trade-off is done in accordance with the law and the SSS technically allow to fully automise the process and it enhances the provided information in two ways: One, the authenticity of the answers can be evaluated by each controller and can be retained should it be questioned by third-parties. Two, the data subject due to delegated request is provided with a more complete picture of the flow of her / his personal data in her / his concrete instance of the business process.

[1048] How this key distribution is accomplished is out of scope.

18.6. Concept: Verifiable and transparent database unions of data distributed in the cloud.

Distributed storage of data base records is one application of the cloud computing paradigm, e.g. “Database as a Service” [253]. Reasons to split database records are manifold; e.g., for efficiency large databases can be split and distributed over several database servers. For reasons of data protection during storage, data can be split into personal parts and parts that contain data, which on its own is anonymous. Splitting the data allows to store them in clouds with different security requirements.

In more detail, each single field of each record in a complete database is signed as an element using a redactable and mergeable signature scheme, which allows records or databases to be split by redaction.

EXAMPLE

To clarify assume the following example: The example depicted in Fig. 145 shows a simple database with one table and three columns (`id`, `Name`, `DNA`), as a very rough example of a bio data bank. Those data banks are collections of samples of human bodily substances (e.g., DNA) that are associated with personal data. The example was already used in the joint publication with *K. Samelin* published at ACNS 2014 is given (see Appendix A publication n^{o} 20) [387]. When this database is initially signed, one gets one signature over the complete database. The example assumes that a private, updatable and mergeable RSS is used for signing each single field of each row in the table as a block of the RSS. Now, one can split the complete record into one table containing only (`id, Name`) pairs, and a second one holding (`id, DNA`) pairs. To do so, one generates two separate signatures by using the redact operation of the RSS. When using a secure and private RSS, then after being signed, also the split databases retain a valid signature. Clients can request records from a server holding a split database and check the integrity of the results by verification of the signature. Here, the RSS is further used by each database server to generate and hand over a valid signature to clients for any elements returned by their queries. If a query only returns one row, the database server redacts all other rows and sends the resulting signature alongside the row to the client.

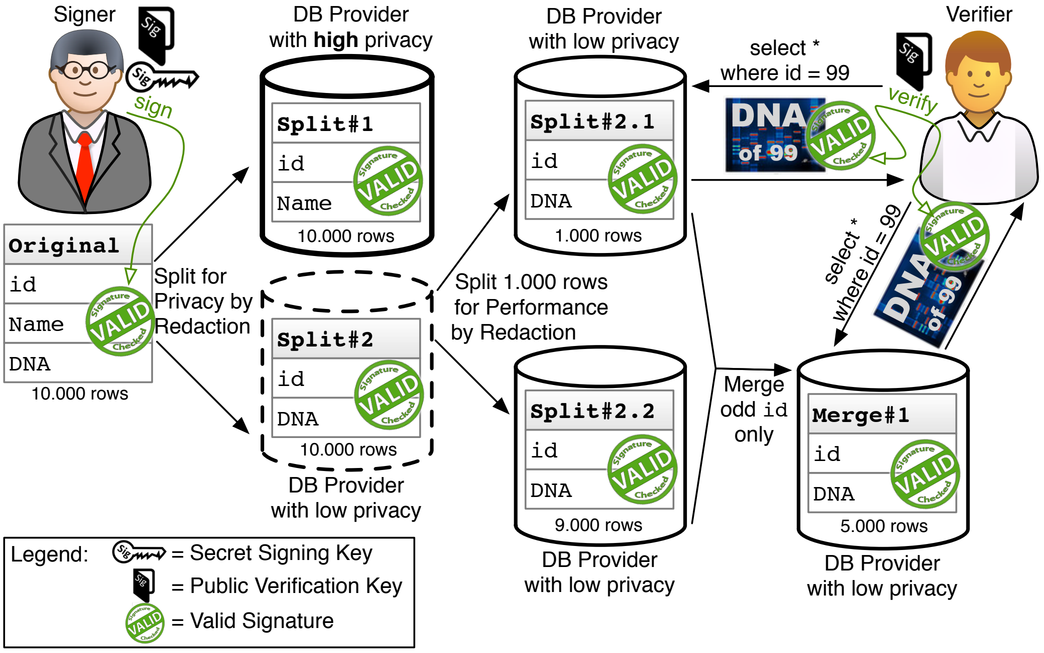


Figure 145. Example of splitting a redactable signed database table and keeping verifiable integrity

Further, this allows separating the records such that highly confidential values are not stored on servers with an insufficient security clearance. While the personal data part might be quite small, the medical data is often very large. However, from the association with personal data follows the necessity to obey strict data protection requirements. Once this association is removed, the protection can be (partly) forgo and the handling of this data becomes simpler[1049]. This motivates to store tables containing columns

[1049] Note, this is an assumption. If the handling is actually simplified by this step needs to be legally analysed for this individual scenario. Especially, it has neither been analysed for this scenario in this thesis nor the joint publication.

with personally identifying information separate from those that contain other data, which on its own is anonymous. Splitting the data allows to store it on servers with different security requirements, and possibly different costs. Also, the outsourcing party wants to allow the database service provider to re-partition the database as it sees suitable for an optimal operation.

In general, the fulfilment of the notion of standard privacy allows to make sure that no information about redacted elements can be derived from signatures, *if one has no access to them.* In this scenario, this means that the low-security servers gain *no* information about the high-security columns.

Mergeability allows combining database records later. If single records are split it allows each single database server to combine several records in one answer. The recipient can verify their signature. Moreover, the client can ask several database servers if he has the needed clearance and merge their answers and gain assurance that they come from the same source and establish that the records have not been modified. Due to the scheme's privacy the parts that have been removed during splitting cannot be reconstructed without having access to them. Hence, parts without a need for a high-level of protection can be stored on servers with lower security requirements.

Additionally, a third cloud service can carry out the **merging** step. Its level of trust depends on the data it merges. Due to merge transparency, even the information that an answer has been created through a merge remains unknown to the recipient. In all cases, the correctness of returned data stored and merged can still be verified. The client is able to verify that the result set is from the original signer and the returned values have not been modified. In turn, mergeability now allows deriving a *single* valid signature from formerly split database records. Hence, the database service provider can undo the splitting by merging the result set, and the signatures, before returning it. The property named *merge transparency* (see Definition 210 in Sect. 14.2.4) even hides the information that a signature has been created through a merge operation from the verifying client.

If the splitting was done for privacy, any client or third-party service which is allowed to query for private data, e.g., (`id, Name`), and (`id, DNA`) can merge the database server's answers and gain assurance that the merged result containing data with three columns (`id, Name, DNA`) comes from the trusted source and establish that the records have not been modified. On the other hand, dynamic updates have been motivated in previous work before [40, 281] and allow the signer, i.e., the database owner, to dynamically add new records to the database.

In the application scenario sketched above, the RSS is required to guarantee that:

- an attacker cannot generate non-legitimate signatures,
- redacted information remains hidden, and
- that all algorithms create *linkable* versions of the same document.

The scenario especially highlights that the third requirement makes RSSs useable in practice: An attacker must not be able to generate "clones" of a signed set by gradually redacting different elements. This application is only possible if the 'mergeability' is formally captured within the scheme and if one is sure that the scheme is able to keep that under the consented control of the Signer, i.e., Signer defines through the scheme and the signature under which circumstances signatures can be merged into a single one.

EXAMPLE

As a brief example, assume that `Name` consists of several columns (`FirstName`, `MiddleName`, `FamilyName`). Then, one does not want that the DNA record of "Rose" "Fitzgerald" "Kennedy" can be duplicated without detection into several sets by redaction of some of the elements, e.g., pretending two signed DNA records exist: One for a member of the "Kennedy" family and another one for a woman with the first name of "Rose".

Merging thus serves as a test when its questionable whether or not two records are derived from the same original: once two signed sets are available, their signatures only merge, if they are linkable. With $merge\mathcal{RSS}$ this thesis introduced such a formal definition of the inverse operation of redaction, thereby this thesis extended the state of the art, e.g. the work done in [314]. As stated before these examples and results have been published [387] (see Appendix A publication n^{o} 20).

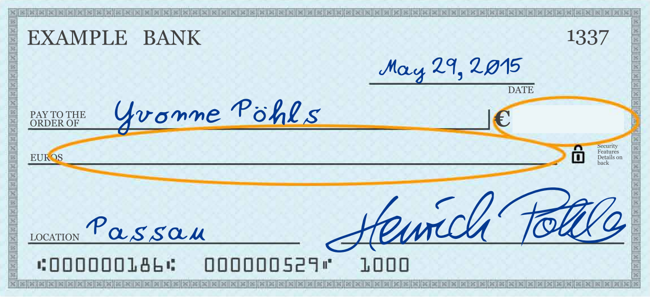
18.7. Concept: Electronic blanket statements

This thesis had discussed the legal implications of blanket statements in Sec. 4.5.5. In the following this thesis shows how an SSS that provides a high probative value can — also following [224] — indeed be used to technically construct such blanket statements. The legal implications of delegating signing rights have been discussed within the German Regulations[1050]. The finding was that a legal blanket statement can be created by the signatory when allowing and consenting to subsequent authorized modifications to the signatory's signed document. The concept of a sanitizable signature scheme allows the Sanitizer to modify only those blocks of data that have been authorized by the Signer by specification of ADM. Hence, the thesis argues that legal delegation can be technically supported by using an secure and private SSS with a high probative value.

EXAMPLE

For this example consider that each block of a document is a field in a form. Namely, consider the form to be an electronic version of the old paper-based bank cheque as illustrated in Fig. 146. Assume further that the signatory uses a malleable signature scheme that would allow a Sanitizer to modify a field, i.e., a sanitizable signature scheme is needed in this example. Then the Signer could allow sanitization of a certain form field by a Sanitizer. This Sanitizer then becomes the delegatee who could modify the data in that form field. The Signer signs all form fields using a malleable signature.

Note, the mere possibility of potential future modifications by the delegatee would be visible from the signature scheme, even if the sanitizable form fields could not be distinguished from unchangeable form fields. Blank cheques are one example of what is legally called a blanket statement in this thesis [361]. Legally, the signer is allowed to leave certain fields underspecified or empty, allowing them to be filled with information later. If done in a consented way, then any specific information filled in later is attributed to the original signer of a blanket form (see Sec. 4.5.5).



EXAMPLE BANK

1337

May 29, 2015

DATE

PAY TO THE ORDER OF Yvonne Pöhls

€

EUROS

Security Features Details on back

LOCATION Passau

Heinrich Pöhls

⑆000000186⑆ 000000529⑈ 1000

Figure 146. Bank cheque with the amount made out in blank: Example of a form to be filled-in later as some fields have no values written in it (circled just for highlighting); as the entire cheque is already signed it is assumed to create a blanket statement

Then, depending on the malleable signature scheme it would be possible or impossible for the Verifier to identify whether an authorized modification has taken place by the delegatee or not (see also Sec. 3.10.2 for an introduction to non-interactively publicly accountable and transparent schemes). Changed in an authorized way or not, but not modified unauthorized, the Signer's signature is verifiable by the Verifier. As per definition of an MSS the Signer stays identifiable via valid signature verification under his public verification key as the originator of all derived documents that can be generated by authorized modifications from the original signed document.

Initially, the Signer could have left a field of the form, represented as one block, empty and allow sanitization of that block by a Sanitizer. This Sanitizer is then the delegatee who could fill in the empty field, like in a blank paper cheque as shown in Fig. 146. The entry of actual values for sanitizable fields into the signed form is delegated to the delegatee, while the delegator still signed the data. This shows the concept of how to construct an electronic blanket statement using a sanitizable signature scheme's modification policy ADM as the "letter of authorization" found in § 172 BGB. The construction clearly captures the signer's consent following the argumentation leading to Evaluation Result 4[1051]. Regarding the economic value of the blanket statements or any signature generated by an MSS for which the Verifier knows that an authorized subsequent modification has actually occurred (Case 3 or 4 as discussed in Subsections 17.5.2.4 and 17.5.2.5). Economically the risk could be acceptable, if the signatory is

[1050] Other regulatory regimes might handle this differently, but this is out of scope.

[1051] Evaluation Result 4: A signature with an MSS with ACA – ≥1CD – PUB integrity but no Sanitizer-accountability generates a legal blanket statement (see Sec. 17.9).

delegating signing rights to a delegatee that the Verifier can get compensation for damages from later. This requires that (1) the Sanitizer's identity can be corroborated from the blanket statement by the Verifier in order to identify the legal entity and (2) that the Sanitizer is technically accountable (Case 3 as discussed in Sec. 17.5.2.4). The application of blockwise non-interactive public accountability may allow to identify the actual risk better as the Verifier knows which information is original and which was subject to a subsequent authorized modification.

18.8. Definitional: Contingency – the dual of integrity

The material of the following section about contingency has been published in the proceedings of IFIPTM 2013 [379] (see Appendix A publication n^{o} 14) and in a talk given at the ISC 2012 [378].

In 2009 the "dual of integrity" [421] termed *contingency* [33] got introduced by *Rost and Pfitzmann*. Here, 'dual' means that something can be proven to be in only one of two states, in this case data either has "integrity" or it is "contingent". Hence, *contingency* describes the verifiable state that the data's integrity is intended to be unknown. In this section this thesis revisits this interesting privacy property, which is in line with Pfitzmann's long history of privacy research [190].

In particular, this thesis is interested if it can be securely achieved if signing rights are cryptographically verifiable delegated, like it is achieved by certain MSS. This thesis argues that a delegation will gradually weaken integrity protection, which generates integrity protection of weaker strengths. From this follows a need to re-evaluate the trust the Verifier can put on information under weaker or no integrity protection. Hence, this section also investigates the legal implications of contingency.

18.8.1. Motivating example for contingency in information and communication technology (ICT)

Contingency is introduced as a method to increase privacy.

EXAMPLE

As an example *Rost and Pfitzmann* mention the following legal case [421]: It is the right of a female employee to answer the employer's question if she is pregnant with "No". Even if pregnant at the time of her statement, she shall face no consequences shall the fact become known later. Hence, she has the right to revise a previously given statement without the revision being held against her. The case shall be handled as if she modified her answer to "Yes" in all records that would have shown her lying [469].

Technically this means that revisions of this answer must not be detected. When revisions go undetected, then even if the statement is still verifiable to have originated from her, one cannot hold her technically responsible and hence one cannot construct legal consequences upon the technically integrity-protected record.

Note, with contingency the loss of integrity protection that is clearly happening does not directly impact origin authentication, which is important when analysing possible legal consequences.

18.8.2. Digital signatures protection of integrity

Integrity can be protected by several means, e.g., access control, message authentication codes or digital signatures. This thesis focuses on the standalone and a-posteriori verifiable integrity protection mechanisms using digital signatures. A verifiable digital signature can fulfil the following technical sub-requirements of contingency protection mechanisms:

- to capture consent,
- offer the requested origin authentication, and
- a legal assessment can be based on the existing signature regulations and the digital signature's property of non-repudiation.

MSS as described in this thesis can be used as they describe a concept in which "the signer delegates the signing rights of parts of the message to a designated party, the sanitizer."[12]. The delegation of the right to generate a valid digital signature on behalf of the signer is a well studied concept in cryptography; other cryptographic solutions are proxy signatures, as introduced by *Mambo et al.* in [328], or group signatures, as introduced by *Chaum and van Heyst* [121].[1052]

18.8.3. Levels of weakened integrity leading up to contingency

As introduced in Sec. 6.2.3 using the Fig. 34 standard non-delegated digital signature offers very strong integrity protection, as every subsequent change is detected, i.e., NCA integrity is offered. Such protection would be at the right spectrum of Fig. 34.

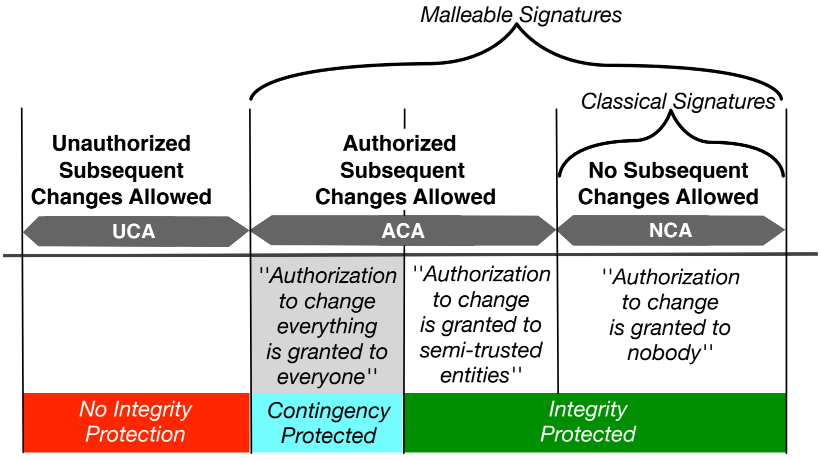

Malleable Signatures
Classical Signatures
Unauthorized Subsequent Changes Allowed
Authorized Subsequent Changes Allowed
No Subsequent Changes Allowed
UCA
ACA
NCA
"Authorization to change everything is granted to everyone"
"Authorization to change is granted to semi-trusted entities"
"Authorization to change is granted to nobody"
No Integrity Protection
Contingency Protected
Integrity Protected

Figure 34. Malleable signature schemes (MSS) and classical digital signature schemes (CDSS) regarding the first aspect of the extended integrity definition: Three levels (UCA, ACA, and NCA) of increasing strictness for tolerable subsequent modifications (same Figure as on page 144)

The detection of every subsequent modification is legally important (see the analysis results in Sec. 4.6.4[1053]as well as the resulting Requirement 3[1054]). Hence, unforgeable digital signature schemes combined with other organisational[1055] and technical measures[1056] fulfil legal requirements set out by EU signature legislation. However, when weakening the NCA integrity protection, without making it reach UCA — the left most of Fig. 34 — the integrity protection falls into ACA as authorized subsequent modifications are allowed. In the most extreme case of ACA everyone can sanitize the full message, which renders the integrity protection verifiably inexistent. Hence, one achieves contingency [421], the "dual of integrity" [33].

The property of contingency itself might at first sight look unintuitive, undesirable and not worth pursuing. As an extreme corner case, however, it is useful to discuss how much integrity protection is needed to fulfil legal and application specific requirements. According to *Rost and Pfitzmann*, the first obvious technical solution ensuring contingency would be **not** to employ integrity protecting mechanisms [421] in the first place. However, not applying integrity protection must be done **intentionally**. It must be a conscious decision not to apply mechanisms, which allow integrity violations to be detected [421]. This intention must be verifiable to allow a precise legal analysis. Hence, a naive technical solution that just does **not** digitally sign, will **not** provide contingency.

[1052] Indeed, from a high level of abstraction proxy signatures [327, 328] are a related concept. However, they allow for delegating signing rights entirely, while sanitizable signatures allow for altering a specific signed message. Hence, their goals differ and thus proxy signatures are not discussed in this thesis.

[1053] Legally this is long standing from the previous European Union's Directive 1999/93/EC on a Community Framework for Electronic Signatures which required "that any subsequent change of the data is detectable" [175] to the current Regulation 910/2014 which states the same in Regulation 910/2014 Article 26 (d) Regulation 910/2014.

[1054] Requirement 3 (Mechanism MUST allow for the verification of integrity and authentication of origin; both SHOULD be non-interactive public to ease the verification) (see Sec. 7.3).

[1055] e.g., public key must be bound to legal entity, as done by public key infrastructures

[1056] e.g., signature must be created by a secure signature creation device, like a secure smart card

Regarding accountability note that even in the case of contingent data the Signer remains identifiable as the origin of the data by means of the public signature verification key $\mathsf{pk}_{\mathsf{sig}}$ in which the Verifier trusts. Quite contrary to accountability of origin, the data's integrity is verifiably not known to the Verifier. It is exactly this contradiction that this section will further dissect with respect to the legal implications and issues of trust. Obviously, it allows reducing the initially very strong integrity protection by allowing others to sign on behalf of the Signer.

This is the first secure technical solution achieving contingency and was published in the proceedings of IFIPTM 2013 [379] (see Appendix A publication n^o 14). Neither, the original work by *Rost and Pfitzmann* [421], nor the same authors [420] nor closely related work [33] give a more precise definition or a precise technical construction.

18.8.4. A definition of contingency or intervenability

This thesis has defined integrity on data as follows:

Definition 113 : Data integrity

Data integrity is a specific state of data that is verifiable (potentially by a third party).
A verification of data integrity that yields a ***valid*** *output corroborates that the integrity-protected data has* ***not*** *been modified (or destroyed) in an unauthorized manner* ***since***
the integrity protection mechanism has been applied to the data.
A verification of data integrity that yields an ***invalid*** *outcome corroborates that the integrity-protected data has been modified (or destroyed) in an unauthorized manner* ***after***
the integrity protection mechanism has been applied to the data.

Notes for Definition 113:

Note 1 Definition 113 does not exclude authorized modifications.

Note 2 Definition 113 requires to precisely capture in an integrity policy (e.g., ISO 10181-6 [340]) what constitutes an unauthorized or authorized change for this data. Compliance with this integrity policy is implicitly assumed to be checked to the full extent by the integrity verification mechanism.

Note 3 Definition 113 sees data integrity in accordance with ISO 10181-6 as "a specific invariant on data" [340]. Following this, the integrity protection mechanism "detects the violation of internal consistency" [340]. "A datum is internally consistent if and only if all modifications of this item satisfy the relevant integrity security policies." [340]

Note 4 Definition 113 is not concerned with the external consistency (e.g., as in [125]), or veracity [215] of the data.

Note 5 Definition 113 is hinting that integrity should allow the detection of integrity violations by third parties. However, it is generic in this respect and thus the level of detection (as defined in Sec. 6.3) may vary, and it depends on the technical detection achieved by the mechanism.

Note 6 Definition 113 explicitly leaves it open to define if the current state of integrity can be checked by a third party or not with an additional property.

Rost and Pfitzmann defined the "dual of integrity" [421] to be the verifiable state of data of not having integrity. But, duality means more than just not having integrity. Duality in this case means that information can only be in the state of *integrity* or in the state of *contingency*, but data cannot be in both, as this would be a contradiction. The work by *Rost and Pfitzmann* from 2009, available in German, called it "Kontingenz" [421], which *Bedner and Ackermann* translated into "contingency" [33]. Later, the term contingency is also called *intervenability* by *Rost and Bock* in [420]. This English version of the German article [419] describes contingency as the data subject's "ability to intervene" [420]. *Data subject* is the term for "an identified or identifiable natural person" given in the European data protection law in the EU Directive 95/46/EC [174]. The ability to intervene is described as an umbrella term to allow the data subject to exercise the rights given by data protection law. [420] In detail contingency allows "information processing entities [...] to demonstrate that they actually have steering control over their systems and are not dominated by the system" [419]. In accordance with [420] this thesis sees both terms as synonyms and continues to use the original term's translation contingency from [33].

Definition 235 : Contingency:

> ***Contingency*** *describes the verifiable state that the data's integrity is intended to be unknown. Data in that state is said to be* contingent*. Contingency is a verifiable property explicitly established by the applied protection mechanism and not an accidental state.* [1057]

To put it differently: For contingency-protected data, it becomes technically not deductible if the data itself is sincere and thus integrity would hold, or if it is insincere, and thus integrity would be violated. In this context, this thesis uses the terms *sincere* and *insincere* to circumvent using English terms which have already a pre-defined meaning in information security[1058]. Further analysis of related work found that the opposite of integrity is defined differently, e.g., in the information flow community the opposite is confidentiality [50].

The notion of contingency as given by *Rost and Pfitzmann* becomes more clearly related to a general privacy notion when studying their authors' motivation[1059]: The absence of integrity offered by contingency is described as a method to increase privacy in [421]. Following in line, the work of *Bedner and Ackermann* subsumes properties like *plausible deniability* under the umbrella notion of contingency in [33]. *Rost and Pfitzmann* also mention the special legal case where one has the right to lie, in other words, one has the right to revise a previously given statement without it being detected. From the fact that revisions go undetected follows that one cannot held responsible and hence one faces no consequences.

18.8.5. Applications and legal implications

This section illustrates legal consequences using two possible applications of contingency protection.

18.8.5.1. "Right to lie"

This is the one example for use of contingency given by *Rost and Pfitzmann* [421]. Legally, there is actually no right to lie. In most cases, e.g., to lie for financial gains, it has legal consequences. Only in exceptional cases there are explicitly no legal consequences of a lie.

EXAMPLE

The example given by *Rost and Pfitzmann* is the right of a female employee to answer the employer's question if she is pregnant with "No". In a job interview a female applicant can give an untruthful answer to the employer's question: "Are you pregnant?". However, she shall not be held accountable for the lie, as a truthful answer "Yes, I am pregnant." could have negative consequences and hence violate the equal treatment for men and women. Hence, even if pregnant at the time of her answer, she shall face no consequences shall the fact become known later. Hence, it is as if she modified her answer in all records to "Yes" [469].

This equal treatment for men and women with regards to access of employment is a codified legal right in the European Economic Community (EEC) [469]. Judges have ruled in favour of lying women[1060] and thereby removed any negative consequences that arose from untruthfully answers to that question.[1061] Following this legal motivation a technical system should allow the user to not always produce integrity-protected data. For the pregnancy example *Rost and Pfitzmann* state that technical systems often are

[1057] For simplicity, this thesis omits the original definition's constraint that this state might only be a temporal state, i.e., only for a given time [421].

[1058] In particular, potentially alternatives of the translations of the German original term "echt" like "genuine", "correct" or "authentic" cause misunderstanding by having pre-occupied meanings.

[1059] Other work of *Pfitzmann* in the area of privacy can be found in [190].

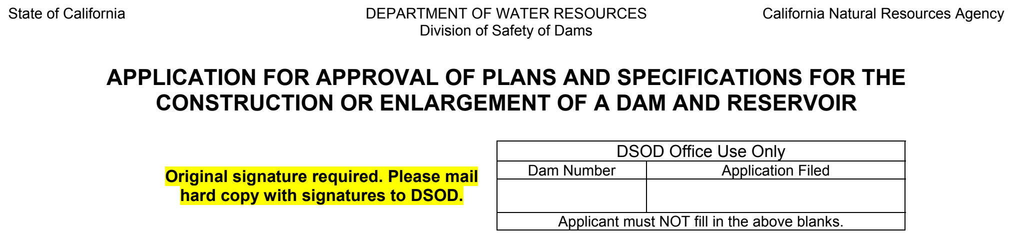
[1060] The European court (Fifth Chamber) stated this in its judgement Wiebke Busch vs. Klinikum Neustadt GmbH & Co. Betriebs-KG. (Case C-320/01) on 27.02.2003.

[1061] Especially Article 2 (1) and (3) of the Directive 76/207/EEC [469] are considering this. In particular, Article 2 (3) mentions "[...] the protection of women, particularly as regards pregnancy [...]" [469].

designed to always answer correctly, e.g., would extract the information about a known pregnancy from the woman's electronic patient record [421]. If the system can be instructed to output contingent data there is no assurance of integrity. However, the visibility of contingency is not negative as it reflects the fact that legally an employee's answer to this question cannot be checked for integrity violations.

18.8.5.2. Underspecified statements

Printed forms can contain fields that are not to be filled in by the applicant but subsequently by administration. For example, forms state "Applicant must not fill in grey fields." [431][1062], another example is shown in Fig. 148 [454]. Other examples are templates of documents that are discussed in the work of *Hanser and Slamanig* [224]. On paper, the applicant fills only the appropriate fields, leaving certain ones empty, and signs the form.



State of California — DEPARTMENT OF WATER RESOURCES — California Natural Resources Agency
Division of Safety of Dams

APPLICATION FOR APPROVAL OF PLANS AND SPECIFICATIONS FOR THE CONSTRUCTION OR ENLARGEMENT OF A DAM AND RESERVOIR

Original signature required. Please mail hard copy with signatures to DSOD.

DSOD Office Use Only	
Dam Number	Application Filed
Applicant must NOT fill in the above blanks.	

Figure 148. Example of a paper based form with fields that are not supposed to be filled in when the first signature is made on the form; excerpt from [454]

Electronically, this quickly becomes complex: If the signer fails to sign empty administrative fields, the administration could add more administrative fields later. If the empty administrative fields are fully integrity-protected by classic signatures, the verification of a completed form fails after the administration has inserted data into the signed previously empty fields. Only after the removal of the administration's content the applicant's signature verifies again, but this requires additional business logic in the verification preparation process.

With weak integrity selectable on message parts the applicant can sign the form and define his fields as fully integrity-protected and the administration's grey fields as contingent. As a second step the administration could then use the supplied sanitizer key to fill in the fields. This would remove the need of the verification procedure to remove/exclude the administration's input. Administration shall finally sign the whole application form with an integrity preserving signature to prohibit further changes by third parties. By using a contingent signature scheme for the grey fields, the applicant explicitly acknowledges that he or she knew about the unfilled grey parts. Additionally, the applicant limits the administration to changing only contingent grey fields, while integrity-protected fields must be left unchanged, to keep the applicant's signature valid.

18.8.6. Towards constructions for contingency protection

Following the work of *Rost, Pfitzmann and Bock* [420, 421] this thesis postulates that a technical contingency solution has to fulfil all of the following requirements:

a) The integrity of contingent information cannot be established.

b) The absence of integrity on contingent information cannot be established.

c) The state is not an error state, but explicit and verifiable by third parties.

Note,

d) Contingency cannot be achieved if no consent is given.

e) Contingency cannot be achieved if it is not indistinguishable from an error, i.e., an invalid digital signature is not achieving contingency.

[1062] Unfortunately, the resource [431] from 2009 used in the original publication is no longer online.

Considering point ‘d’ from above requires that protection of contingency must be applied explicitly and consented. Point ‘e’ from above is also in the direction of point ‘d’, but explicitly states that contingency is not an error condition of integrity. But the last point explicitly calls for technical constructions that must make subsequent modifications indistinguishable.

Note, this thesis is aware that a straw man construction of contingency can be achieved if the signed part contains the verifiable statement of consent that the other half has indeed not been signed and is not original and that this was done on purpose. This is highlighted with two examples:

For the first example assume a system would sign one half of data with a digital signature to gain integrity protection and just not sign the other half. As the other half is not protected, anyone can add or remove information without any consent of the signer; further the second half’s state can not be verified. While the first half being signed creates integrity protected data. However, being unsigned does not create contingent data for the second half of the data.

EXAMPLE

For the second example assume the system would sign the complete data to protect integrity and then the system would change something on purpose to violate the integrity protection, i.e., the system invalidate its own signature.

EXAMPLE

The construction in the above example allows to make any further subsequent modifications indistinguishable. However, this would not contain a verifiable consent. It can also be achieved be achieved if done following the idea in the above example, i.e., first half signed containing a statement of consent, second half signed, but invalidated on purpose.

Both examples indicate possible technical instantiations that achieve indistinguishability and the consent for contingency. Looking more closely at ways to technically instantiate contingency this thesis recognised that their internals come close to those of some of the MSS presented in this thesis. For example the construction given for $pubacc\mathcal{SSS}$ (see Sec. 13.4) with two partially overlapping signatures and a chain of verification keys, would leave after a sanitization blocks of the message to verify under one verification key, while other blocks would not verify under this key. Hence, contingency is modelled using an SSS as an underlying building block. The construction of a provably scheme achieving contingency and a more rigorous game based definition of contingency is given next (Sec. 18.8.7). Elegantly simple the cryptographic solution achieves contingency securely.

18.8.7. Construction of a contingent signature scheme using an SSS

The construction is built upon a transparent, and hence private, SSS. The Signer creates both key pairs and holds sk_{sig}, pk_{sig}, as well as sk_{san}, pk_{san}. The construction attaches the Sanitizer’s secret key sk_{san} and distributes it alongside the message. Using the sanitizable signature scheme the construction signs an extended message k:

$$k = \mathsf{sk}_{\text{san}}||\mathsf{pk}_{\text{san}}||\mathsf{pk}_{\text{sig}}||m[1]||m[2]||\ldots||m[\ell-1]||m[\ell]\,.$$

Every block of the original m of length ℓ is marked admissible such that the complete m is sanitizable for holders of sk_{san}. Only the keys sk_{san}, pk_{san} and pk_{sig}[1063] attached to the front of the extended message k are immutable. Hence, every recipient of the message signature pair (k, σ) including the Signer, could possibly modify m. This means that the Signer explicitly and verifiably denied the integrity protection for the message, namely for all message blocks that comprise m. Note, with $\ell = 1$ the construction would not need to think of the message as a composition of message blocks. m is contingent, because the transparent SSS used prohibits third parties receiving k from detecting if k is original or sanitized.

Construction 16 : Contingency Signature Scheme (*contingent$\mathcal{SS}$*)

Let SSS be a secure and transparent sanitizable signature scheme that consists of the following efficient (PPT) algorithms: SSS $:= (\mathsf{KGen}_{sig}, \mathsf{KGen}_{san}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Sanitize}, \mathsf{Proof}, \mathsf{Judge})$.

Then define contingent$\mathcal{SS}$ $:= (\mathsf{CKeyGen}, \mathsf{CProtect}, \mathsf{CVerify}, \mathsf{CChange})$ *and implement it as follows:*

[1063] pk_{san} could be derivable from sk_{san}, but for readability the scheme simply distributes both.

Key Generation: *On input of the security parameter, the algorithm* $\mathsf{CKeyGen}$ *returns both key-pairs running* KGen_{sig} *and* KGen_{san}.

Protect contingency: *The algorithm* $\mathsf{CProtect}$ *on input of the message* m *and the* Signer*'s secret signature generation key* sk_{sig} *and the public key* pk_{san} *works as follows:*

```
To ensure public sanitization of $m$ only the three keys are immutable:
        $\mathsf{ADM} = (4,5,6,\ldots,\ell,\ell+1,\ell+2)$
Build an extended message, with the sanitizable $m_1$ to $m_\ell$
        $k = \mathsf{sk}_{san}||\mathsf{pk}_{san}||\mathsf{pk}_{sig}||m_1||m_2||\ldots||m_\ell$
Return as output: $(k,\sigma) \leftarrow \mathsf{Sign}(1^\lambda, k, \mathsf{sk}_{sig}, \mathsf{pk}_{san}, \mathsf{ADM})$
The extended message and signature will be distributed as a pair $(k,\sigma)$.
```

Verification of contingency: $\mathsf{CVerify}$ *on input of k,* σ *and* pk_{sig} *works as follows:*

```
Extract the public key $\mathsf{pk}_{san}$ from k.
Verify the signature $\sigma$ on k: $d \leftarrow \mathsf{Verify}(1^\lambda, k, \sigma, \mathsf{pk}_{sig}, \mathsf{pk}_{san})$
    if $d = \mathtt{true}$ return $\mathtt{true}$ for a valid signature,
    else return $\mathtt{false}$ for an invalid signature.
```

Change contingent data : *The algorithm* $\mathsf{CChange}$ *on input of* k *and* σ *by applying the modification defined by* MOD *works as follows:*

```
Extract $\mathsf{sk}_{san}$, $\mathsf{pk}_{sig}$ from k
Calculate a $\mathsf{MOD}^k$ from the supplied $\mathsf{MOD}$ such that
    it modifies the message $m$ inside k.
Generate a new signature $\sigma'$ and modify k to $k'$:
    $(k',\sigma') \leftarrow \mathsf{Sanit}(1^\lambda, k, \mathsf{MOD}^k, \sigma, \mathsf{pk}_{sig}, \mathsf{sk}_{san})$
Verify the signature: $d \leftarrow \mathsf{CVerify}(1^\lambda, k', \sigma')$
    if $d = \mathtt{false}$ return $(k',\sigma')$,
    else return $\perp$.
```

Note, ADM can be reconstructed from σ and sk_{san} can be reconstructed from k simply by reversing the concatenation. Note further that the construction does not require to previously generate a key pair $(\mathsf{sk}_{\text{san}}, \mathsf{pk}_{\text{san}})$, CProtect could generate it for each run on the fly. However, without loss of generality the parameter is kept to visualise the similarity to the SSS construction.

18.8.8. Security of the contingency construction

Above Construction 16 (*contingent*$\mathcal{SS}$) is secure iff the underlying SSS is unforgeable, immutable, private, transparent and accountable. To instantiate this new scheme one can facilitate the sanitizable signature scheme based on chameleon hash-functions proposed by *Brzuska et al.* [64], it provably offers all the required security properties. This section sketches the proofs of security with respect to contingency and integrity.

18.8.8.1. Integrity of keys

Integrity protection is only required for the keys. They should not be removable from the contingent message.

Proof 54 : Sketch

The adversary has access to a Signing-Oracle which will generate contingent signatures on supplied messages m_j, *denoted as j pairs* (k_j, σ_j). *To break integrity the adversary needs to change one key of the extended message* k^*, *i.e., one of the first three blocks of* k^* *must be different from previously queried pairs. The following two properties must hold:*

(a) $k_j^*[x] \neq k_j[x]$ *for any j and* $x = 1,2,3$

(b) $\mathtt{true} \leftarrow \mathsf{CVerify}(1^\lambda, k^*, \sigma^*, \mathsf{pk}_{sig})$

Hence, the adversary has two options, either a direct forgery or modify a genuinely signed k_j using the underlying Sanit*. Assume the underlying* SSS *to be unforgeable, then the first option is not possible without breaking immutability of the* SSS*. k includes only m as admissible parts, hence the adversary cannot modify the keys either. Thus, an adversary breaking the scheme's integrity protection also breaks the immutability or unforgeability of the underlying* SSS*.* □

18.8.8.2. Contingency of the message

The informal definition of contingency from Definition 235 has no indication of computational limitations of the party trying to make the distinction. As with some of the SSS properties one can define an unconditional contingency property (see Definition 236) for practical purposes with computationally bounded parties: Contingency protection requires that an adversary cannot distinguish between the original or a subsequently modified message. Any subsequent change to a contingent message would still result in a contingent message. The adversary has only access to a Signing-Oracle, which will generate a contingency protection for the supplied message.

Definition 236 : Contingency:

Data is is said to be ***contingent****, if for any efficient (PPT) adversary $\mathcal{A}$, the probability that the game depicted in Fig. 149, returns 1 is negligibly close to $\frac{1}{2}$ (as a function of λ).*

```
**Experiment** $\mathsf{Contingency}_{\mathcal{A}}(\lambda)$
$(\mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}}, \mathsf{sk}_{\mathrm{san}}) \leftarrow \mathsf{CKeyGen}(1^\lambda)$
$b \overset{\$}{\leftarrow} \{0,1\}$
$a \leftarrow \mathcal{A}^{\mathsf{CSign}(1^\lambda,\mathsf{sk},\cdot),\mathsf{TouchedOrUntouched}(1^\lambda,\ldots,\mathsf{sk}_{\mathrm{sig}},b)}(1^\lambda, \mathsf{pk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
    where oracle TouchedOrUntouched for input $m$ and MOD:
        $(k,\sigma) \leftarrow \mathsf{CProtect}(1^\lambda, m, \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
        $(k',\sigma') \leftarrow \mathsf{CChange}(1^\lambda, k, \mathsf{MOD}, \sigma, \mathsf{pk}_{\mathrm{sig}}, \mathsf{sk}_{\mathrm{san}})$
        if $b = 1$
            extract $m'$ from $k'$
            $(k',\sigma') \leftarrow \mathsf{CProtect}(1^\lambda, m', \mathsf{sk}_{\mathrm{sig}}, \mathsf{pk}_{\mathrm{san}})$
        return $(k',\sigma')$.
return 1, if $b = a$
```

Figure 149. Contingency Experiment

Proof 55

Noticeably, the experiment for contingency in Fig. 149 resembles that of transparency from Fig. 53 on page 268. Assume the underlying SSS *to be private, transparent and immutable, then the sanitizer is able to change any part of the message m, which is included in k. From the adversary winning the contingency experiment from Fig. 149 an adversary on the transparency experiment obviously follows. Thus, an adversary breaking the proposed scheme's contingency protection will also break the transparency of the underlying* SSS*.* □

18.8.9. Accountability and integrity mixed with contingency

The algorithm Judge from the underlying SSS only works if the Signer is actively participating using Proof that needs $\mathsf{sk}_{\mathrm{sig}}$. If the Signer is not participating it is usually assumed that the Signer is the origin. Contingency protection is a conscious decision by the Signer, it did not allow integrity violations to be detected [421]. However, if the Signer would participate in the interactive protocol of accountability from the SSS it could indeed remove contingency by giving out correct proofs using Proof. The above Construction 16 could easily be adapted to cater for messages that contain blocks which are either contingent or integrity-protected. This is achieved by excluding integrity-protected blocks of m from ADM. An adversary cannot change immutable blocks if the underlying SSS is secure, if changed it would result in an invalid signature and hence every subsequent modification is detected as required for integrity protection.

19 — Conclusion, contributions and future work

19.1. Conclusion

After a thorough analysis of the legal demands for confidentiality and integrity protection they are formulated as technical requirements in Part I and II of the thesis. It is now possible to precisely-enough identify the aspects in which malleable signature schemes (MSS) differ from classical legally accepted schemes (CDSS) by virtue of the refined integrity notion given in Part I (Chapter 6). New cryptographic properties presented in Part III minimise the technical difference and address shortcomings identified in existing works with respect to security as well as practicality and flexibility. Part III concludes with new constructions that provably achieve the requested properties in allowing those MSS (Chapters 13 and 14). Finally, Part IV evaluates that they satisfy the legal demands (Chapter 17) and thus those MSSs become cryptographic solutions that are applicable in a broad range of scenarios (Chapter 18). As a result, the research question can be answered positively.

The research question is:

Can a malleable signature scheme be private to be compliant with EU data protection regulation **and at the same time** fulfil the integrity protection legally required in the EU to achieve a high probative value for the data signed?

The answer is:

Yes, if the scheme:

- offers standard privacy following *Brzuska et al.*, and
- offers ACA – ≥1CD – PUB integrity, and
- offers a cryptographic strength against unforgeability that follows from provably secure algorithms with state-of-the-practise security parameters, and
- allows treating the underlying asymmetric keys following the needed organisational requirements, in particular this requires that the scheme
  - allows to issue qualified certificate for the public key(s) from trust service providers that are accredited with respect to Regulation 910/2014, and
  - executes inside a hardware security module to protect the secrecy of signature generation key(s).

Schemes to achieve this:

The sanitizable signature scheme $pubacc\mathcal{SSS}$ (see Sections 13.4 and 13.5) and the redactable signature scheme $pubacc\mathcal{RSS}$ (see Sections 14.13 and 14.14) presented in this thesis satisfy all the above:

- Both offer standard privacy defined in accordance with *Brzuska et al.* (Subsections 11.6.3 and 11.13.3).
- Both achieve the required integrity protection of non-interactive public accountability (ACA – ≥1CD – PUB as introduced in Sec. 6.4.3 and defined for SSS in Sec. 13.1 and for RSS in Sec. 14.1.
- Both can be instantiated upon primitives for which rigorous security proofs exist (see provable security in Sec. 2.3.1) and with legally accepted security parameters (see Subsections 7.7.2 and 17.3.7).
- Both signature primitives are asymmetric and standardised algorithm algorithms for which the public keys can get certified within the existing public key infrastructures.
- Both have been prototypically implemented on commercially available secure smart cards (see Sec. 14.15 and Sec. 13.10.2).

19.2. Main results and contributions of this thesis

In summary this thesis

- harmonises cryptographic properties and devises new or refines existing properties (Sec. 19.2.1),
- extends the integrity notion to capture important differences between sanitizable signature schemes (SSSs), redactable signature schemes (RSSs), and classic signature schemes (CDSS) — as well as among different constructions within each class (Sec. 19.2.2),
- formally separates sanitizable (SSSs) and redactable signature schemes (RSSs) (Sec. 19.2.3),
- aligns legally accepted classic signature schemes (CDSS) and malleable signature schemes (MSS): remaining difference is the permission of authorized subsequent changes due to the proposed cryptographic property of non-interactive public accountability (PUB) and achievement of $\geq$1CDintegrity for MSS (Sec. 19.2.4),
- legally evaluates the proposed schemes and shows that selected RSS and SSS offer a high probative value (Sec. 19.2.5),
- constructs RSS and SSS based on standard or known cryptographic primitives that are proven to achieve the proposed cryptographic properties (Sec. 19.2.6), and
- constructs from an SSS a protection mechanism of contingency, which is the dual of integrity as introduced by *Rost and Pfitzmann* [421], and formerly defines contingency, which can be used to facilitate personal data protection (Sec. 19.2.7).

These main results and contributions get briefly summarised in the remainder of this section.

19.2.1. Harmonisation of existing cryptographic properties and design of new or refined properties

This thesis presents the security properties of the current state of the art for RSS and SSS in a harmonised notation (see Chapter 11). Based on a requirement analysis the thesis then further formally extends the existing properties and proposes new properties (see Chapters 13 and 14) to address shortcomings (see Chapter 12). Among others, it

1. details cryptographic properties to a finer scope, i.e., from message-level to block- or groups-of-blocks-level properties;
2. formally captures differences in the cryptographic properties' strengths — this might look like subtleties, i.e. weak, standard or strengthened properties, but those differences are of paramount importance for the legal evaluation, i.e. public instead of interactive provides easy detectability of any subsequent modification by a Verifier;
3. proposes new properties, in particular the property of non-interactive public accountability, which further helps to formally clarify the existing property of transparency.

For all extensions the thesis provides formal game-based descriptions (as described in Sec. 3.8.1) which allow to rigorously proof security (as defined in Sec. 2.3.1). It also states interesting new relations, or shows that the extension is non-invasive with respect to the often intuitive and well-standing existing relations between cryptographic properties.

This harmonisation requests a more detailed notion of certain integrity aspects than what existing integrity notions had to offer. Thus, an extension of the integrity notion is necessary — which is another result of this thesis (see the following Sec. 19.2.2).

The harmonisation and extensions are offered individually for two classes of MSS: Redactable signature schemes (RSS) and sanitizable signature schemes (SSS). Those are indeed cryptographically two classes as they can be formally separated, which is another contribution (see Sec. 19.2.3).

19.2.2. Extended integrity notion that captures important differences between RSS, SSS and CDSS and within each class

This thesis first generalises the notion of integrity and then extends it with three aspects which explicitly capture the intention towards authorized subsequent modifications, their detection and the achieved accountability. The general basic notion is as follows:

> "Data Integrity is a specific state of data that is verifiable (potentially by a third party). [...]" [Definition 113].

For the focus of this thesis and the resulting understanding of integrity it is important that an integrity protection mechanism

- verifiably establishes the absence of any unauthorized modification for integrity-protected data, but
- leaves the decision what constitutes an unauthorized modification under the full control of the party applying the mechanism.

The thesis finds that it is not uncommon for existing technical integrity notions to allow authorized changes: In some it is even explicitly stated[1064], but more often it is implicit[1065] (see Analysis Result 11[1066]).

To complicate things further, the previously existing terminology is neither suitable to compare different malleable signatures among each other, nor does it allow to identify the properties an MSS needs in order to offer comparable functionality to those of existing legally recognised classical digital signature schemes (CDSS). As a consequence, three aspects are added to the basic notion quoted above. This results in the proposed integrity definition being able to explicitly differentiate them. The three aspects are as follows:

1st: authorization of subsequent changes (see Sec. 6.2)
The aspect of authorization **explicitly** describes the need to protect integrity even in application scenarios in which a subsequent modification to integrity-protected data should not harm the data's integrity. Alongside, this requires to understand and define authorized subsequent modifications to distinguish them from unauthorized modifications. The extended notion of integrity models this aspect and differentiates between ACA and NCA — authorized changes allowed (ACA) vs. no authorized changes allowed (NCA). The notion of ACA integrity now explicitly states the need for support from the integrity protection to hold even if subsequent but authorized modifications are present. Previously, this was mostly[1067] only implicitly contained in the integrity definition.

2nd: Detection of subsequent changes (see Sec. 6.3)
The aspect of detection of occurred unauthorized and authorized modifications **explicitly** captures what subsequent or future modifications can be detected and by whom. This clearly maps to the legal requirements and serves as a differentiator between existing and newly proposed schemes. It is the detectability of an absence of any subsequent change that is demanded for an increased probative value according to the the legal analysis. Naturally, the proposed notion of integrity and accountability is devised to capture this and denotes it as $\geq$1CD integrity. This makes it possible to denote integrity protection with legally required detection capabilities while extending it further to allow authorized subsequent modifications: ACA – $\geq$1CD integrity.

3rd: Accountability for subsequent changes (see Sec. 6.4)
Last, but of paramount importance for the legal value of a signature scheme is the ability to gain accountability. Here, the integrity notion is extended to describe to which extent the Verifier is enabled to use the cryptographic mechanism to produce technical evidence to hold a party accountable. The result of this

[1064] As examples take the works of *Biba* (see Sec. 5.1.1), *Bishop* (see Sec. 5.1.9) and *Clark and Wilson* (see Sec. 5.1.3) where this co-notation is explicitly present.

[1065] For an example, ISO 7498's definition of integrity in the reference model for open systems interconnection (OSI) [239] includes authorized modifications implicitly: "Data integrity is the property that data has not been altered or destroyed in an unauthorized manner" [239].

[1066] Analysis Result 11: Existing technical definitions of integrity differ in their permission of authorized subsequent modifications; some are not explicit in this respect (see Sec. 5.5.4).

[1067] See for example the overview in Tab. 1 on page 137.

thesis is to — at least — differentiate accountability towards a third party with technical non-repudiation into two forms: public non-interactive (PUB) and non-public interactive (INT). The finding is that electronically signed data is given a high probative value if it achieves ≥1CD – PUB integrity (see Sec. 17.6).

Enriched with these three aspects the proposed integrity notion allows to explicitly capture the mandatory and recommended legal requirements: NCA / ACA – ≥1CD – PUB is recommended. The proposed notion is also general enough to describe differences in existing integrity definitions (see Sec. 6.8). Further it allows to classify the protection offered by existing signature schemes including the existing MSS schemes:

A classical digital signatures scheme (CDSS) based on secure cryptographic hashes, e.g., RSASSA-PSS from PKCS-v2.2 [422] offers NCA – 1CD – PUB integrity: This means Verifiers detect any subsequent modification without distinguishing between authorized or unauthorized modification and verification proves the Signer's accountability for unmodified messages non-interactively, i.e., by using just the Signer's public verification key, the proof of origin can be verified.

The schemes $pubacc\mathcal{SSS}$ **and** $pubacc\mathcal{RSS}$ offer ACA – 1CD – PUB: Like for the CDSS above, Verifiers detect any subsequent modification. Of course they allow to authorize subsequent modifications (ACA). Their public accountability (PUB) together with their detection of any subsequent modification (1CD) allows to determine if the document is original (unchanged), sanitized or redacted (Sanitizer is accountable) or maliciously modified (signature is invalid).

Contrary, an unlinkable and transparent sanitizable signature scheme by *Brzuska et al.* [67] that is Signer-accountable offers ACA – 1CD – INT integrity: Obviously, the SSS allows to authorize subsequent modifications (ACA), but due to the property of transparency it requires a correctly-executed and non-repudiable interaction (INT) to gain accountability. Still, that SSS offers accountability and allows detection of any subsequent modification (1CD).

19.2.3. Formal separation of SSS and RSS

This thesis formally separates sanitizable (SSS) and redactable signature schemes (RSS) and shows that they are two separate, yet both powerful classes of special signature schemes that both achieve integrity with authorized subsequent modifications. The redaction functionality of the redactable signature schemes (RSS) used in this thesis follows the descriptions given by *Steinfeld, Bull, and Zheng* [455] and *Johnson, Molnar, Song, and Wagner* [273]. RSS follow a different concept when compared with the functionality offered by sanitizable signature schemes (SSS) following the operational descriptions by *Ateniese, Chou, de Medeiros, and Tsudik* [12]. In a nutshell, an RSS allows to remove a block while an SSS allows to override a block for which the authorization was given by the Signer (see Sec. 3.4 for initial background).

Of course RSS and SSS share the same common high-level goals: Both allow an entity in the role of the Signer to express its authorization for a controlled weakening of the integrity protection by means of the signature generation. Hence, both concepts allow to authorize subsequent modifications that will not invalidate the Signer's digital signature. The separation of RSS and SSS (see Chapter 15) shows that a black-box construction of an RSS from an SSS will yield only a weakly private RSS, e.g. one where the Verifier will always be able to see redactions, e.g., as special symbols like ■. Even a strongly private SSS will only result in an RSS that is weakly private, but not strongly. There are however several efficient RSS schemes where this information indeed remains hidden. This means that both RSS and SSS instantiations have their own merits because of the different functionality they offer efficiently. In the end, an application's demand, i.e., redaction or arbitrary subsequent modification, decides which class to deploy.

19.2.4. Alignment of CDSS and MSS by proposing the property of non-interactive public accountability (PUB) and ≥1CD integrity

The proposed extended definition introduces a legally necessary distinguishing aspect to the integrity notion: non-interactive public accountability. The thesis finds that current technical and legal definitions of the term "Data Integrity" neither adequately, nor precisely enough define this facet. In this respect, the extended integrity notion explicitly includes two aspects. It first describes in different levels the integrity protection mechanism's ability to offer the Verifier the detection of the subsequent changes, not limited to the detection of integrity violations. This is required to further discuss the notion of accountability for malleable signature schemes, especially based on Analysis Result 12[1068].

Digital signature schemes offer a kind of accountability which is advantageous as it includes technical evidence against a repudiation of signature generation[1069]. As this thesis tries to generate evidence towards untrusted third-parties, this advantage is especially important. The second helpful aspect included in the proposed extended definition of integrity differentiates explicitly which form of accountability is reached. Here, the notion of non-interactive public accountability (PUB) and its cryptographic achievement is one very central contribution.

As discussed the legally accepted classical digital signature schemes (CDSS) achieve NCA – 1CD – PUB integrity protection (see the visualisation in Sec. 6.10). In order for MSS to come close in functionality to those legally accepted CDSS it is necessary to clearly capture — and provably achieve — the possibility of Verifiers to identify the accountable party without further interaction and without knowledge of secrets, or ACA – ≥1CD – PUB integrity in short. When achieving ACA – ≥1CD – PUB integrity protection it is possible to argue that this increases the probative value (see Sec. 17.6). This is another contribution (see Sec. 19.2.5).

Important in the context of the quest to offer the public form of accountability (PUB) is that with the achievement of PUB accountability the opposing cryptographic property of transparency can no longer be achieved. In this respect, it is especially interesting that a negation of the existing property of transparency does not yield the desired public form. Conveniently, while transparency is stronger, i.e., *transparency* $\Rightarrow$ *privacy*, it is not necessary for privacy: $\neg$ *transparency* $\not\Leftarrow$ $\neg$ *privacy* (see Theorem 2 on page 273). With the formal definition of the cryptographic property of non-interactive public accountability this thesis solved the cryptographic challenge to carefully remove the cryptographic transparency while not negatively impacting privacy.

To reach the goal of a strong analogy between CDSS and MSS, the thesis consequently suggests (marked by SHOULD) for the Signer, and hence for the legal signatory, to choose a suitable scheme or control the accountability for subsequent modifications in order to maintain technical equivalence with existing cryptographic algorithms for qualified electronic signatures. With non-interactive public accountability, the identification function[1070] [147, 149] can be provided by the mechanism regardless of the cooperation of the Signer or the Sanitizer (see Sec. 4.5.1 for a list of the signature functionalities).[1071]

All evaluations and observations are based on the analysis of classical digital signature schemes that are currently judged as being legally compliant. The analysis found that non-interactive public accountability (PUB) is not an explicitly required in any legal or technical definition that was analysed (see also Sec. 6.8 for an overview of the accountability of existing technical definitions). However, schemes like RSASSA-PSS from PKCS-v2.2 [422] provide 3^{rd}-party non-interactive public accountability (see Sec. 6.10) which is in line with the notions of public verifiability and transferability (see

[1068] Analysis Result 12: Definition of accountability for the current version of the validly signed, but subsequently modified document (see Sec. 5.5.5 on page 138).

[1069] This is opposed to the integrity offerings by message authentication codes (MACs) based symmetric keys. ISO 10181-6 states this main advantage of digital signatures over the other integrity mechanisms as follows: "Digital Signatures permit the set of entities which can validate the data to be arbitrary in size and composition." [340]. On the contrary a symmetric-key-based MAC has the problem that "[...] entities capable of sealing the data and the set of entities capable of validating the data are [...] coincident [...]" [340].

[1070] In German it is called 'Identitätsfunktion' [147, 149].

[1071] The opportunity to achieve this signature functionality for schemes that are transparent, i.e., interactive non-public accountability (INT), as defined in Definition 109, is diminished. A signature scheme where authorized subsequent modifications stay hidden from the Verifier is not easily found technically equivalent to legally accepted schemes and thus might not offer the same increased probative value.

Definition 101).[1072] This is also in line with the stated functionality that a hand-written signature can be checked by comparing it to another one of known authenticity using handwriting examination[1073]without the need for an involvement of the signatory as postulated in the functional description of signatures from German legal text [149]. Hence, this thesis strongly recommends the achievement of this public accountability, as indicated by the word SHOULD in Requirement 3.3[1074]. Authentication SHOULD be verifiable in a non-interactive manner, as only then it can be clearly argued that it is 'easy' or 'simple' for the Verifier to carry out the identification function[1075].

To summarise, if an RSS or SSS achieves ACA – ≥1CD – PUB then the difference between it and a legally accepted signature scheme, which achieves NCA – ≥1CD – PUB, is reduced to authorized modifications being allowed (ACA) or not (NCA). Obviously contrary to a CDSS a malleable signature scheme allows authorization of subsequent changes (ACA). However, malleable signature schemes equally allow to reliably detect any subsequent modification may it be authorized or unauthorized (≥1CD). As well, Malleable signature schemes as well allow the Verifier to identify the accountable party of a validly signed message from just public knowledge and without further interactions (PUB). Hence, this thesis achieves to make this gap as small as necessary which aids the evaluation of the probative value that an MSS can offer.

Finally note that the cryptographic notion of accountability, as defined for SSSs in [64] was not even containing the public form. This cryptographic security property is formally specified in this work (see Sections 6.4 and 13.1 and Sec. 11.6.6) and is published in the joint work with *C. Brzuska* and *K. Samelin* [68] (see Appendix A publication n° 11). Alongside, constructions are proposed that provably achieve[1076] this security property, e.g. $pubacc\mathcal{SSS}$ (see Sec. 13.5). In summary, non-interactive public accountability has not been defined for RSSs previously and is formally described in Sec. 14.1 and published as joint work with *K. Samelin* [388] (see Appendix A publication n° 21).

19.2.5. Legal evaluation yields that MSS can award an increased probative value

The thesis identifies the technical requirements for integrity and accountability that increase the legal probative value based upon an analysis of the main legal texts in the EU for integrity and authenticity protection. One should note that legal texts describe the outcomes in high-level language, which allows them to stay technologically neutral[1077] (see also Analysis Result 3.3[1078]). As a result this work offers a technically precise set of ten requirements including two for the confidentiality[1079] distilled from related legal texts and court rulings. Both, the requirement extraction, as well as the attribution of a high probative value for malleable signatures that fulfil all needed mandatory requirements, builds on the following assumption stated by *Roßnagel*:

[1072] Also schemes like RSASSA-PKCS-v1.5-SIGN [274] provide this form of accountability, however RSASSA-PKCS-v1.5-SIGN is at the end of its lifetime, following the catalogue it shall only be used until the end of 2017, as discussed in Sec. 5.3.1; RSASSA-PSS from PKCS-v2.2 [422] is still listed as legally accepted beyond 2017 [85].

[1073] Translated from the German original text it states: "Die Identität kann im Streitfall z. B. durch einen Unterschriftenvergleich verifiziert werden." [149].

[1074] Requirement 3.3 (Mechanism SHOULD achieve non-interactive public verification of integrity, i.e., ≥1CD–PUB integrity) (see Sec. 7.3.3 on page 172).

[1075] In German it states "[...] kann der Adressat in **einfacher Weise** überprüfen, ob die Signatur **mit dem privaten Signaturschlüssel des Schlüssel-Inhabers erstellt** worden ist und ob der signierte Text **nachträglich verändert worden** ist." [147]; bold face has been added for highlighting.

[1076] This section uses the notion of 'provably secure' in the currently common cryptographic meaning of "reductionist security" [402] as detailed in Sec. 2.3.1.

[1077] To the extreme that legal texts describe functionality usually termed by the technical property of integrity while the term 'integrity' itself is absent in legal texts, e.g., in the European Regulation on Electronic Signatures [175].

[1078] Analysis Result 3.3: Electronic signature legislation texts describe required functionality in technology-neutral language (in Sec. 4.6.3.3 on page 104).

[1079] Requirement 9 (Mechanism MUST prohibit recovery of information from the sanitized or redacted data from the adjusted signature with legally accepted state-of-practice strength) in Sec. 10.1 and Requirement 10 (Mechanism MAY prohibit the identification of additional context information further to Requirement 9 without violating Requirement 4) in Sec. 10.2 are towards the goal of confidentiality.

> "The checking assumes that the prerequisites for and the requirements of qualified electronic signatures and their legal consequences are meant to reference each other; thus they need to be be understood and interpreted as linked even if found in different legal acts."[1080] [410]

Hence, a high probative value, as attributed by legal norms to qualified electronic signatures, and with it the statutory evidentiary presumption will only be given if all the technical and organisational requirements and pre-requisites set forth in all concerning legal acts are fulfilled altogether at once. One of the technical and organisational requirements supporting the cryptographic signature scheme is to enable the signatory to stay in control of the secret signature generation key. Current solutions for CDSS do this by use of a hardware security module (HSM), legally Regulation 910/2014 (eIDAS) is requiring for a so-called qualified signature-creation device (QSCD)[1081]. This thesis shows with prototype implementations of RSSs and SSSs that selected MSS schemes can indeed be executed on a commercially available secure smart card (see Sections 13.10 and 14.15).

With all other technical and organisational requirements set forth either directly by law or by referenced technical recommendation, e.g., specifications of government bodies (like BSI [75, 466], ENISA [188], or the EU [170]) are met, the signatures created become **qualified** electronic signatures in accordance with Regulation 910/2014. Only then they are "legally equivalent to hand-written signatures" [Regulation 910/2014, Directive 1999/93/EC]. And only then documents signed with aforementioned signatures offer a high value of evidence, e.g., are at least considered as prima facie evidence in court. The Evaluation Result 3[1082] and Evaluation Result 4[1083] discuss this in more detail. In addition, documents signed by an MSS with an increased probative value become economically more valuable than unsigned documents as they offer evidence for future legal disputes.

19.2.6. New RSS and SSS constructions provide standard privacy and an increased probative value

As the answer to the research question is yes, the thesis shows that a standard private MSS can offer ACA – ≥1CD – PUB integrity; and thus achieves both goals simultaneously. Namely, the five constructions $pubacc\mathcal{SSS}$, $pubacc\mathcal{SSS}^{Block}$ (see Sec. 13.5 for both constructions), $blockacc\mathcal{SSS}^{PUB}$ (see Sec. 13.7.2.2), $unlinkable\mathcal{SSS}$ (see Sec. 13.9), and $pubacc\mathcal{RSS}$ (see Sec. 14.14) protect the confidentiality by standard privacy — defined in accordance with *Brzuska et al.* (Subsections 11.6.3 and 11.13.3) — while offering the legally recommended integrity protection and accountability — ACA – ≥1CD – PUB integrity. Thus, these private malleable signatures allow balancing two opposing goals: On the one hand, they mitigate privacy issues by allowing to modify data, such that the previous version or even the position of modification stays confidential. On the other hand, they still keep a legally required level of integrity and authenticity by allowing a Verifier to identify that the unmodified data is original and, if modified in an authorized way, that the Sanitizer is accountable and the authorisation was given with the consent of the original signatory, who can still be identified by means of the Signer's public key. With this 'twin pack' of properties the constructions retain the most important cryptographic properties of classic digital signatures and thus this thesis argues that they inherit their protection and offer a high probative value for the data signed by them, while they still allow for authorized changes such that they achieve confidentiality for the original blocks of the message.

Without that ability for subsequent authorized modifications by MSS a trusted third party is needed to retain the authenticity of a modified authenticity-protected document: For evidence generation purposes the fully authentic and non-redacted document is given to a trusted third party. After verification of authenticity the trusted third party would carry out the intended modification(s). The third party would thus be trusted for two actions: (1) having correctly verified the authenticity; and (2) having correctly carried out only authorized redactions or sanitizations. It must be trusted by both entities — by the

[1080] The original German text states: "Dabei geht die Prüfung davon aus, dass Voraussetzungen für, Anforderungen an und Rechtsfolgen von qualifizierten Signaturen aufeinander bezogen sind und in diesem Bezug aufeinander verstanden und interpretiert werden müssen, auch wenn die Regelungen in unterschiedlichen Gesetzen zu finden sind." [410].

[1081] Also know from the previous Directive 1999/93/EC as an secure signature-creation device (SSCD).

[1082] Evaluation Result 3: A signature by an MSS with ACA – ≥1CD – PUB, that yields that the signed document is unmodified, is a qualified electronic signature following Regulation 910/2014 (see Sec. 17.6).

[1083] Evaluation Result 4: A signature with an MSS with ACA – ≥1CD –PUB integrity but no Sanitizer-accountability generates a legal blanket statement (see Sec. 17.9).

Verifier and by the Signer: The Signer, as the disclosing party, entrusts the third party with the full document. The verifying party needs to trust the third party with the correct verification of the initial document and the correct redaction or sanitization. RSS and SSS allow to relief the Sanitizer from being trusted to do only authorized sanitizations (or redactions) of a validly signed document[1084]. With RSS or SSS the valid verification outcome of a signature generated by them implicitly verifies that the Sanitizer did work on valid inputs from the Signer and within the boundaries declared by the Signer.

In total this thesis presents ten different constructions, grouped into eight schemes. In particular, integrity protection must achieve the detection of all **un**authorized changes (NCA – ≥UCD or ACA – ≥UCD[1085]). None the less important for an increased probative value is that the signature scheme allows to provide accountability. Public accountability and accountability of the Sanitizer, as achieved by the five constructions $pubacc\mathcal{SSS}$, $pubacc\mathcal{SSS}^{Block}$, $blockacc\mathcal{SSS}^{PUB}$, $unlinkable\mathcal{SSS}$ and $pubacc\mathcal{RSS}$. It is not directly mandated by the laws and regulations within the EU to achieve this. However, these constructions provide ACA – ≥1CD –PUB and are thus closest to the NCA – 1CD – PUB integrity provided by the classic and legally accepted schemes. This makes these five constructions ranked the highest when ranked by their direct fulfilment of mandatory and recommended requirements as shown in Tab. 38.

Rank	Proposed Construction	Integrity offer	Section
TOP	$pubacc\mathcal{SSS}$	ACA – 1CD – PUB	13.5.1.1
TOP	$pubacc\mathcal{SSS}^{Block}$	ACA – BCD – PUB	13.5.1.2
TOP	$blockacc\mathcal{SSS}^{PUB}$	ACA – ≥1CD,< BCD – PUB	13.7.2.2
TOP	$pubacc\mathcal{RSS}$	ACA – 1CD – PUB	14.14.1
TOP	$unlinkable\mathcal{SSS}$	ACA – 1CD – PUB	13.9
MID	$blockacc\mathcal{SSS}^{INT}$	ACA – ≥1CD,< BCD – INT	13.7.2.1
The following RSS do require to be combined with an SSS according to the ARSS framework. If combined with an ACA – 1CD – PUB SSS then they are equally top-ranked as the $pubacc\mathcal{RSS}$ construction, but retain their additional redaction flexibilities.			
	$priv\mathcal{RSS}$	ACA – UCD	14.6.2
	$flex\mathcal{RSS}$	ACA – UCD	14.8.2
	$struct\mathcal{RSS}$	ACA – UCD	14.10.2
	$merge\mathcal{RSS}$	ACA – UCD	14.12.2

Table 38. Ranking of proposed constructions according to their direct fulfilment of all recommended and mandatory requirements

If one removes the need for public accountability (PUB), but keeps the requirement for interactive non-public accountability (INT) the remaining five constructions from this thesis are potentially suitable as they give ACA – ≥UCD integrity. Potentially, because accountability of at least INT is needed. ACA – ≥1CD – INT is offered by one remaining SSS construction: $blockacc\mathcal{SSS}^{INT}$, but not directly by four proposed RSS constructions. The former is marked as medium ranked (MID) in Tab. 38, the latter remain unranked. $blockacc\mathcal{SSS}^{INT}$ (see Sec. 13.6 for the concept) allows to increase the scope of detectability and offers interactive non-public accountability (INT) for grouped blocks. It allows to adjust the overhead that comes with an increased visibility of occurred authorized modifications on blocks, only for the needed groups rather than for all. However, without PUB accountability the construction is a bit further away from the functionality of the legally accepted schemes and arguably provide less legal certainty as discussed in Sec. 6.4.2.

The four RSS schemes are transparent with a public redaction functionality as usually found in RSS, hence they completely lack Sanitizer-accountability: $priv\mathcal{RSS}$ (see Sec. 14.6.2), $flex\mathcal{RSS}$ (see Sec. 14.8.2), $struct\mathcal{RSS}$ (see Sec. 14.10), $merge\mathcal{RSS}$ (see Sec. 14.12). While the four RSS do not achieve ≥1CD – PUB on their own, the thesis shows how to provably gain it in combination with a secure SSS following the proposed framework for an accountable redactable signature scheme (ARSS) as given in Sec. 14.1. As an ARSS they offer their additional RSS features and are Sanitizer-accountable.

[1084] Hence, the Sanitizer is declared a semi-trusted party.

[1085] That is equal to NCA – ≥1CD or ACA – ≥UCD, see note in Sec. 6.3.1 on page 146.

This thesis advances the state of the art by constructing private malleable signatures that fulfil the legally-rooted requirements and hence those schemes could be granted *statutory evidentiary presumption* within the realm of the EU electronic signature laws [Regulation 910/2014, Directive 1999/93/EC]. Electronic signatures generated by schemes that fulfil those requirements would be considered "legally equivalent to hand-written signatures" and offer a legally high probative value. Hence, the computational effort spent on the malleable signature scheme also becomes economically valuable. At the same time, the original data can be kept private by controllable sanitizations to achieve legal data protection requirements. Technically, this keeps the legally desired cryptographic property of being private while allowing properties such as non-interactive public accountability (PUB). Especially, for all newly devised schemes the privacy is kept on at least the same level as the widely accepted state-of-the-art property of standard privacy as defined by *Brzuska et al.* [64] (see Subsections 11.6.3 and 11.13.3) as shown in Evaluation Result 1[1086].

19.2.7. SSS technically protect the dual of integrity: Contingency

With contingency, a term for the *dual of integrity* introduced by *Rost and Pfitzmann* [421], data can be proven to not have integrity. Due to duality, data can be only in one of two states, in this case data either has "integrity" or it is "contingent". Hence, contingency describes the verifiable state that the data's integrity is intended to be unknown. This thesis cryptographically constructs contingency protection from an SSS (see Sec. 18.8). To the best of the author's knowledge this is the first provably secure[1087] construction of this — not obvious — data protection concept of contingency. The proposed construction nicely allows mixing integrity and contingency protection under one signature. It is published at the IFIP Trust Management conference (IFIPTM 2013) [379] (see Appendix A publication n° 14) and was presented at the Information Security Conference (ISC 2012) [378]. One could obviously argue that a verifier should not trust a message, which is not integrity-protected. However, if the sender is left with the option to apply integrity protection or not, i.e., sign the message or not, then the receipt of an unsigned message allows at least two interpretations: Either the signer did not sign, or a third party removed the signature. The latter can be technically achieved without being detected. Applying contingency protection is hence like answering the question "*Do you really want no integrity protection?*" with a clearly audible "*No integrity, thanks.*" Receiving contingent data allows the receiver to verifiably identify the signer's consent to the absence of integrity. This evidence can also be presented and verified by third parties.

19.3. Future work and open problems

The following selected areas have been mentioned as out of scope, however they can be considered for future work[1088].

Formalising the levels of the accountability axis
The two forms of the accountability axis (PUB and INT) that are introduced generally in Sec. 6.4 are very useful to this thesis and the difference makes a legal impact, at least in terms of legal certainty. The difference is equally of technical importance, as the two forms differ in their communication overhead, PUB doe not need to communicate with the Signer or a secret key holder. This advantage might be important in scenarios of high-latency networks or networks where communication is costly like the Internet-of-Things.

For a formal exploration it is interesting to note that the two forms of accountability, i.e. INT and PUB, do by themselves not form a lattice. As said, the two forms clearly describe a different form of accountability. Furthermore, they can be clearly separated as mutually exclusive by their informal definitions (cmp. Definition 109 on page 150 with Definition 112 on page 153). It is left as future work to see if such a definitional framework can be given for accountability, or if this is impossible. Introducing a level of no accountability like the theoretical infimum seems intuitive and possible for the aspect of accountability. Still, it is not easily possible to order PUB and INT as currently defined. The

[1086] Evaluation Result 1: All proposed schemes from Chapters 13 and 14 fulfil data protection requirement (Req. 9) as they achieve standard privacy (see Sec. 17.2).

[1087] As defined in Sec. 2.3.1.

[1088] On a personal note, if you are interested to jointly work in one of these directions, please feel free to contact the author of this thesis.

work done by *Pfitzmann* [372] can be a starting point for a formalisation. Namely, it already offers a description and formal model for a "[...] [s]trong requirement of the recipient on disputes." [372][1089] in combination with "Strong requirement of the signer on disputes" [372][1090]. Both are strengthened versions of dispute resolutions.

Legal implications of designated or restricted verifiability
Having a designated verifiability is marked as out of scope of this thesis (see Sec. 2.10.2.3 on page 35). However, it might be interesting to investigate if such a reduced verifiability as offered by designated verifier signature schemes, e.g. [313, 456, 497] in general has any legal consequences. Moreover this property can also be reached for RSS as presented by *Derler et al.* [145]. This thesis assumes 3^{rd}-party verifiability when analysing the probative value, thus to evaluate the probative value offered by designated verifier schemes remains as future works.

Extension of properties and schemes towards the multi-Sanitizer framework
This work limits itself to assume that there is always only one entity in the role of Signer and only one entity in the role of the Sanitizer. Naturally, all the proposed cryptographic notions and schemes of this thesis are formally defined and cryptographically proven secure in this single Sanitizer framework only. It is left as future work to extend the security properties to multiple Sanitizers, as it was already done for the core properties in the multi-sanitizer framework presented by *Canard, Jambert, and Lescuyer* [112]. From the legal analysis of this thesis it follows that a public key used for verification needs to be linked to exactly one legal entity, e.g. a person. However, the analysis does not explicitly disallow multiple entities, e.g., multiple Sanitizers or even Signers, as long as those entities can still be assumed to be distinguishable by individual secret keys. Especially for schemes that do not offer the property of cryptographic transparency and where the list of all potential sanitizers is not considered confidential the extension to multiple sanitizers is assumed to be straightforward (see Sec. 7.4.3). However, it is still future work for other properties and introduces new properties, e.g., privacy of potential Sanitizers' identities.

Legal implications of other special signature schemes
Finally, there are many other interesting specialised signature schemes for which the methods for legal evaluation laid out in this thesis might be applicable to. Like the works of *Sorge* [451] who analysed the legal impact of identity based signatures, it is interesting to know the legal implications created by using a signature scheme. In this regards, the thesis' methodology has already been applied to functional signature schemes. The positive results, which are analog to those for MSS were part of the joint publication with *F.W.J. van Geelkerken* and *S. Fischer-Hübner* [199] (see Appendix A publication n^{o} 22). Other schemes of interest could be designated verifier signatures as mentioned earlier or group-signatures. Both do not consider a weakening of the integrity protection, but impact the detection capabilities.

The End

[1089] "Strong requirement of the recipient on disputes. The recipient wishes that any message he has accepted should also be accepted by a third party in a subsequent dispute." [372].

[1090] "Strong requirement of the signer on disputes. [...] In this case, one need not find the signer to carry out a fair dispute." [372].

20 Bibliography

[1] C. Adams and S. Lloyd. *Understanding PKI: Concepts, Standards, and Deployment Considerations*. Addison-Wesley Professional, 2nd edition, 2002. ISBN 978-0672323911.

[2] N. Adrian. *Modellierung eines "Farm-to-Fork-Modells" als Ansatz zur Durchführung einer gesundheitsökonomischen Analyse: am Beispiel von Maßnahmen zur Senkung der Prävalenz bestimmter Salmonella-Serotypen in den Beständen von Schlachtschweinen*. PhD thesis, Landwirtschaftlich-Gärtnerischen Fakultät Humboldt-Universität zu Berlin, 2012. URL `http://www.bfr.bund.de/cm/350/modellierung-eines-farm-to-fork-modells-als-ansatz-zur-durchfuehrung-einer-gesundheitsoekonomischen-analyse.pdf` [last accessed: Sep. 2017].

[3] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order-Preserving Encryption for Numeric Data. In *Proc. of ACM conference on managment of data (SIGMOD 2004)*, pages 563–574. ACM, 2004. URL `http://doi.acm.org/10.1145/1007568.1007632` [last accessed: Sep. 2017].

[4] S. Agrawal, S. Kumar, A. Shareef, and P. C. Rangan. Sanitizable Signatures with Strong Transparency in the Standard Model. In *Proc. of International Conference on Information Security and Cryptology (INSCRYPT 2009)*, volume 6151 of *LNCS*. Springer, 2010. URL `https://doi.org/10.1007/978-3-642-16342-5_7` [last accessed: Sep. 2017].

[5] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, abhi shelat, and B. Waters. Computing on authenticated data. *IACR Cryptology ePrint Archive*, 096. 2011. URL `https://eprint.iacr.org/2011/096.pdf` [last accessed: Sep. 2017].

[6] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, abhi shelat, and B. Waters. Computing on Authenticated Data. In *Proc. of. Theory of Cryptography (TCC 2012)*, volume 7194 of *LNCS*, pages 1–20. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-28914-9_1` [last accessed: Sep. 2017].

[7] J. H. Ahn, D. Boneh, J. Camenisch, S. Hohenberger, abhi shelat, and B. Waters. Computing on Authenticated Data. *Journal of Cryptology*, 28(2):351–395. Springer, 2015. URL `http://dx.doi.org/10.1007/s00145-014-9182-0` [last accessed: Sep. 2017].

[8] A. M. Al-Khouri. *Critical Insights from a Practitioner Mindset*, volume 3. Chartridge Books Oxford, 2013. ISBN 978-1909287594.

[9] A. Alaqra, S. Fischer-Hübner, J. S. Pettersson, F. van Geelkerken, E. Wästlund, M. Volkamer, T. Länger, and H. C. Pöhls. PRISMACLOUD public deliverable D2.1: Legal, Social and HCI Requirements. URL `https://prismacloud.eu/wp-content/uploads/2017/03/D2.1-Legal-social-and-HCI-requirements_v1.2.pdf` [last accessed: Sep. 2017], 2015.

[10] M. U. Arshad, A. Kundu, E. Bertino, K. Madhavan, and A. Ghafoor. Security of graph data: hashing schemes and definitions. In *Proc. of ACM conference on Data and application security and privacy (CODASPY '14)*, pages 223–234. ACM, 2014. URL `http://doi.acm.org/10.1145/2557547.2557564` [last accessed: Sep. 2017].

[11] G. Ateniese and B. de Medeiros. Identity-Based Chameleon Hash and Applications. In *Proc. of Financial Cryptography 2004*, volume 3110 of *LNCS*, pages 164–180. Springer, 2004. URL `https://doi.org/10.1007/978-3-540-27809-2_19` [last accessed: Sep. 2017].

[12] G. Ateniese, D. H. Chou, B. de Medeiros, and G. Tsudik. Sanitizable Signatures. In *Proc. of European Symposium on Research in Computer Security (ESORICS 2005)*, volume 3679 of *LNCS*, pages 159–177. Springer, 2005. URL `http://dx.doi.org/10.1007/11555827_10` [last accessed: Sep. 2017].

[13] G. Ateniese, B. Magri, D. Venturi, and E. R. Andrade. Redactable Blockchain - or - Rewriting History in Bitcoin and Friends. *IACR Cryptology ePrint Archive*, 757. 2016. URL `https://eprint.iacr.org/2016/757.pdf` [last accessed: Sep. 2017].

[14] G. Ateniese, B. Magri, D. Venturi, and E. Andrade. Redactable Blockchain - or - Rewriting History in Bitcoin and Friends. In *IEEE European Symposium on Security and Privacy (S&P 2017)*, pages 111–126, 2017.

[15] N. Attrapadung, B. Libert, and T. Peters. Computing on Authenticated Data: New Privacy Definitions and Constructions. In *Proc. of Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 367–385. Springer, 2012. URL `https://doi.org/10.1007/978-3-642-34961-4_23` [last accessed: Sep. 2017].

[16] N. Attrapadung, B. Libert, and T. Peters. Efficient Completely Context-Hiding Quotable and Linearly Homomorphic Signatures. In *Proc. of Public-Key Cryptography (PKC 2013)*, volume 7778 of *LNCS*, pages 386–404. Springer, 2013. URL `https://doi.org/10.1007/978-3-642-36362-7_24` [last accessed: Sep. 2017].

[17] J. Attridge. An overview of hardware security modules. URL `https://www.sans.org/reading-room/whitepapers/vpns/overview-hardware-security-modules-757` [last accessed: Aug. 2016], 2002.

[18] M. Backes, S. Meiser, and D. Schröder. Delegatable Functional Signatures. *IACR Cryptology ePrint Archive*, 408. 2013. URL `http://eprint.iacr.org/2013/408.pdf` [last accessed: Sep. 2017].

[19] M. Backes, Ö. Dagdelen, M. Fischlin, S. Gajek, S. Meiser, and D. Schröder. Operational Signature Schemes. *IACR Cryptology ePrint Archive*, 820. 2014. URL `https://eprint.iacr.org/2014/820.pdf` [last accessed: Sep. 2017].

[20] D. J. Baker. The moral limits of consent as a defense in the criminal law. *New Criminal Law Review: In International and Interdisciplinary Journal*, 12(1):93–121. University of California Press, 2009.

[21] S. A. Baker. Don't Worry Be Happy. URL `http://archive.wired.com/wired/archive/2.06/nsa.clipper_pr.html` [last accessed: 21.6.2014], 1994.

[22] J. Balfanz and P. Laue. Transformation und elektronische Patientenakte. *Datenschutz und Datensicherheit - DuD*, 34(12):815–818. Springer, 2010. URL `https://link.springer.com/article/10.1007/s11623-010-0222-0` [last accessed: Sep. 2017].

[23] F. Bao, R. H. Deng, X. Ding, J. Lai, and Y. Zhao. Hierarchical Identity-Based Chameleon Hash and Its Applications. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2011)*, volume 6715 of *LNCS*, pages 201–219. Springer, 2011. URL `http://dx.doi.org/10.1007/978-3-642-21554-4_12` [last accessed: Sep. 2017].

[24] N. Barić and B. Pfitzmann. Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. In *Proc. of Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *LNCS*, pages 480–494. Springer, 1997. URL `https://doi.org/10.1007/3-540-69053-0_33` [last accessed: Sep. 2017].

[25] E. Barker and A. Roginsky. NIST Special Publication 800-131A Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths, 2011. URL `http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf` [last accessed: Sep. 2017].

[26] M. Bartel, J. Boyer, B. Fox, B. LaMacchia, and E. Simon. XML-Signature Syntax and Processing. W3C Recommendation 12 February 2002, 2002. URL `http://www.w3.org/TR/xmldsig-core/` [last accessed: Sep. 2017].

[27] D. Bauer, D. M. Blough, and D. Cash. Minimal information disclosure with efficiently verifiable credentials. In *Proc. of ACM workshop on Digital identity management*, pages 15–24. ACM, 2008.

[28] D. Bauer, D. M. Blough, and A. Mohan. Redactable signatures on data with dependencies and their application to personal health records. In *Proc. of ACM workshop on Privacy in the electronic society*, pages 91–100. ACM, 2009.

[29] J. Bauer, R. C. Staudemeyer, H. C. Pöhls, and A. Fragkiadakis. ECDSA on things: IoT integrity protection in practise. In *Proc. of Information and Communications Security (ICICS 2016)*, volume 9977 of *LNCS*, pages 3–17. Springer, 2016. URL `http://henrich.poehls.com/papers/2016_Bauer-et-al_ECDSA-on-things_ICICS.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1007/978-3-319-50011-9_1` [last accessed: Sep. 2017].

[30] T. L. Beauchamp. Informed consent: its history, meaning, and present challenges. *Cambridge Quarterly of Healthcare Ethics*, 20(04):515–523. Cambridge Univ Press, 2011.

[31] M. T. Beck, S. Krenn, F.-S. Preiss, and K. Samelin. Practical Signing-Right Revocation. In *Proc. of International Conference on Trust and Trustworthy Computing (TRUST 2016)*, volume 9824 of *LNCS*, pages 21–39. Springer, 2016. URL http://dx.doi.org/10.1007/978-3-319-45572-3_2 [last accessed: Sep. 2017].

[32] M. T. Beck, J. Camenisch, D. Derler, S. Krenn, H. C. Pöhls, K. Samelin, and D. Slamanig. Practical Strongly Invisible and Strongly Accountable Sanitizable Signatures. In *Proc. of Australasian Conference on Information Security and Privacy (ACISP 2017)*, volume 10342 of *LNCS*, pages 437–452. Springer, 2017. URL https://doi.org/10.1007/978-3-319-60055-0_23 [last accessed: Sep. 2017].

[33] M. Bedner and T. Ackermann. Schutzziele der IT-Sicherheit. *Datenschutz und Datensicherheit - DuD*, 34 (5):323–328. Springer, 2010. URL http://dx.doi.org/10.1007/s11623-010-0096-1 [last accessed: Sep. 2017].

[34] M. Bellare. Two-Tier Signatures, Strongly Unforgeable Signatures, and Fiat-Shamir Without Random Oracles. In *Proc. of Public-Key Cryptography (PKC 2007)*, volume 4450 of *LNCS*, pages 201–216. Springer, 2007. URL https://doi.org/10.1007/978-3-540-71677-8_14 [last accessed: Sep. 2017].

[35] M. Bellare and D. Micciancio. A new paradigm for collision-free hashing: incrementality at reduced cost. In *Proc. of Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *LNCS*, pages 163–192. Springer, 1997. URL https://doi.org/10.1007/3-540-69053-0_13 [last accessed: Sep. 2017].

[36] M. Bellare and G. Neven. Transitive signatures: new schemes and proofs. *IEEE Transactions on Information Theory*, 51(6):2133–2151. IEEE, 2005.

[37] M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proc. of 1st ACM conference on Computer and Communications Security (CCS 1993)*, pages 62–73. ACM, 1993.

[38] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Advances in Cryptology – CRYPTO' 93*, pages 232–249. Springer, 1994. URL https://doi.org/10.1007/3-540-48329-2_21 [last accessed: Sep. 2017].

[39] M. Bellare and P. Rogaway. The Exact Security of Digital Signatures - How to Sign with RSA and Rabin. In *Proc. of Advances in Cryptology – EUROCRYPT '96*, volume 1070 of *LNCS*, pages 399–416. Springer, 1996.

[40] M. Bellare, O. Goldreich, and S. Goldwasser. Incremental Cryptography: The Case of Hashing and Signing. In *Proc. of Advances in Cryptology – CRYPTO '94*, volume 950 of *LNCS*, pages 216–233. Springer, 1994. URL http://dx.doi.org/10.1007/3-540-48658-5_22 [last accessed: Sep. 2017].

[41] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In *Annual International Cryptology Conference*, pages 26–45. Springer, 1998.

[42] M. Bellare, D. Hofheinz, and E. Kiltz. Subtleties in the Definition of IND-CCA: When and How Should Challenge Decryption Be Disallowed? *Journal of Cryptology*, 28(1):29–48. Springer, 2015. URL http://dx.doi.org/10.1007/s00145-013-9167-4 [last accessed: Sep. 2017].

[43] J. Benaloh and M. D. Mare. One-way accumulators: A decentralized alternative to digital signatures. In *Proc. of Advances in Cryptology – EUROCRYPT '93*, volume 765 of *LNCS*, pages 274–285. Springer, 1993. URL https://doi.org/10.1007/3-540-48285-7_24 [last accessed: Sep. 2017].

[44] W. Benedek. *Die Welthandelsorganisation (WTO): alle Texte einschließlich GATT (1994), GATS und TRIPS: Textausgabe*. C.H.BECK, 1998. ISBN 978-3406434914.

[45] D. Berbecaru and A. Lioy. Exploiting the European Union trusted service status list for certificate validation in STORK: design, implementation, and lessons learnt. *Software: Practice and Experience*, 45(11):1457–1477. Wiley Online Library, 2015.

[46] C. Berger. Beweisführung mit elektronischen Dokumenten. *NJW Neue juristische Wochenzeitung*, pages 1016–1020. C.H.BECK, 2005.

[47] K. J. Biba. Integrity considerations for secure computer systems. Technical Report ESD-TR-76-372. Technical report, The Mitre Corporation, Bedford, MA, 1977.

[48] A. Bilzhause, M. Huber, H. C. Pöhls, and K. Samelin. Cryptographically Enforced Four-Eyes Principle. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2016)*, pages 760–767. Conference Publishing Services (CPS), 2016. URL http://henrich.poehls.com/papers/2016_BilzhauseHuberPoehlsSamelin_4EyesPrinciple_ARES_SECPID.pdf [last accessed: Sep. 2017]. URL https://doi.org/10.1109/ARES.2016.28 [last accessed: Sep. 2017].

[49] A. Bilzhause, H. C. Pöhls, and K. Samelin. Position Paper: The Past, Present, and Future of Sanitizable and Redactable Signatures. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2017)*, pages 87:1–87:9. ACM, 2017. URL `http://henrich.poehls.com/papers/2017_BilzhausePoehlsSamelin_ARES17_RSS_SSS_PastPresentFuture.pdf` [last accessed: Sep. 2017]. URL `http://dx.doi.org/10.1145/3098954.3104058` [last accessed: Sep. 2017].

[50] A. Birgisson, A. Russo, and A. Sabelfeld. Unifying facets of information integrity. In *Information Systems Security*, volume 6503 of *LNCS*, pages 48–65. Springer, 2011.

[51] M. Bishop. *Computer Security: Art and Science*. Addison-Wesley Professional, 2002. ISBN 0201440997.

[52] H. C. Black. *Handbook on the Construction and Interpretation of the Laws, with a Chapter on the Interpretation of Judicial Decisions and the Doctrine of Precedents*. The Lawbook Exchange, Ltd., 2011.

[53] G. Blakley. Twenty years of cryptography in the open literature. In *Proc. of IEEE Symposium on Security and Privacy (S&P '99)*, pages 106–107, 1999.

[54] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13 (7):422–426. ACM, 1970.

[55] B. Boemke and B. Ulrici. *BGB Allgemeiner Teil*. Springer, 2013. ISBN 3642016103.

[56] A. Boldyreva and M. Fischlin. Analysis of Random Oracle Instantiation Scenarios for OAEP and Other Practical Schemes. In *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *LNCS*, pages 412–429. Springer, 2005.

[57] D. Boneh and D. M. Freeman. Homomorphic Signatures for Polynomial Functions. In *Proc. of Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 149–168. Springer, 2011.

[58] D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In *Proc. of Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432. Springer, 2003. ISBN 3-540-14039-5.

[59] D. Boneh, G. Segev, and B. Waters. Targeted malleability: Homomorphic encryption for restricted computations. *IACR Cryptology ePrint Archive*, 311. 2011. URL `https://eprint.iacr.org/2011/311.pdf` [last accessed: Sep. 2017].

[60] J. Boyar, D. Chaum, I. Damgård, and T. P. Pedersen. Convertible Undeniable Signatures. In *Proc. of Advances in Cryptology – CRYPTO '90*, number 537 in LNCS, pages 189–205. Springer, 1990.

[61] J. Boyer. Canonical XML V 1.0, 2001. URL `http://www.w3.org/TR/2001/REC-xml-c14n-20010315` [last accessed: Sep. 2017].

[62] S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. RFC 3447, RFC Editor, 1997. URL `http://www.ietf.org/rfc/rfc2119.txt` [last accessed: Sep. 2017].

[63] S. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy.* MIT Press, 2000. ISBN 978-0262526302.

[64] C. Brzuska, M. Fischlin, T. Freudenreich, A. Lehmann, M. Page, J. Schelbert, D. Schröder, and F. Volk. Security of Sanitizable Signatures Revisited. In *Proc. of Public Key Cryptography (PKC 2009)*, volume 5443 of *LNCS*, pages 317–336. Springer, 2009. URL `http://dx.doi.org/10.1007/978-3-642-00468-1_18` [last accessed: Sep. 2017].

[65] C. Brzuska, M. Fischlin, A. Lehmann, and D. Schröder. Sanitizable signatures: how to partially delegate control for authenticated data. In *Proc. of the Special Interest Group on Biometrics and Electronic Signatures (BIOSIG 2009)*, volume 155 of *GI-Edition Lecture Notes in Informatics (LNI)*, pages 117–128. GI, 2009. URL `http://subs.emis.de/LNI/Proceedings/Proceedings155/article2927.html` [last accessed: Sep. 2017].

[66] C. Brzuska, H. Busch, O. Dagdelen, M. Fischlin, M. Franz, S. Katzenbeisser, M. Manulis, C. Onete, A. Peter, B. Poettering, and D. Schröder. Redactable Signatures for Tree-Structured Data: Definitions and Constructions. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2010)*, volume 6123 of *LNCS*, pages 87–104. Springer, 2010. URL `http://portal.acm.org/citation.cfm?id=1894302.1894310` [last accessed: Sep. 2017].

[67] C. Brzuska, M. Fischlin, A. Lehmann, and D. Schroeder. Unlinkability of Sanitizable Signatures. In *Proc. of Public-Key Cryptography (PKC 2010)*, volume 6056 of *LNCS*, pages 444–461. Springer, 2010. URL `http://dx.doi.org/10.1007/978-3-642-13013-7_26` [last accessed: Sep. 2017].

[68] C. Brzuska, H. C. Pöhls, and K. Samelin. Non-Interactive Public Accountability for Sanitizable Signatures. In *Revised Selected Papers of European PKI Workshop: Research and Applications (EuroPKI 2012)*, volume 7868 of *LNCS*, pages 178–193. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-40012-4_12` [last accessed: Sep. 2017].

[69] C. Brzuska, H. C. Pöhls, and K. Samelin. Efficient and Perfectly Unlinkable Sanitizable Signatures without Group Signatures. In *Revised Selected Papers of 10th European PKI Workshop: Research and Applications (EuroPKI 2013)*, volume 8341 of *LNCS*, pages 12–30. Springer, 2013. URL `http://dx.doi.org/10.1007/978-3-642-53997-8_2` [last accessed: Sep. 2017].

[70] S. G. Bugg and H. Simon. *Langenscheidt Fachwörterbuch Recht Englisch*. Verlag Langenscheidt Fachverlag, 2009. ISBN 9783861172406.

[71] L. Bull, P. Stanski, and D. M. Squire. Content extraction signatures using xml digital signatures and custom transforms on-demand. In *Proc. of international conference on World Wide Web*, pages 170–177. ACM, 2003.

[72] Bundesamt für Sicherheit in der Informationstechnik. Profile Profile for Device, Secure Signature Creation and Core, PP SSCD type 3 (key generation on card). prEN 14169-1: 2009, version 1.01, 2009-12, 2009.

[73] Bundesamt für Sicherheit in der Informationstechnik. BSI TR-03109, 2011. URL `https://www.bsi.bund.de/DE/Themen/SmartMeter/TechnRichtlinie/TR_node.html` [last accessed: Sep. 2017].

[74] Bundesamt für Sicherheit in der Informationstechnik. BSI TR-02102-1: Kryptographische Verfahren: Empfehlungen und Schlüssellängen (Version 2014). URL `https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102_pdf.pdf` [last accessed: Jun. 2014], 2014.

[75] Bundesamt für Sicherheit in der Informationstechnik. BSI TR-02102-1: Kryptographische Verfahren: Empfehlungen und Schlüssellängen (Version 2015). URL `https://www.bsi.bund.de/cae/servlet/contentblob/477256/publicationFile/30924/BSI-TR-02102_V1_0_pdf.pdf` [last accessed: Nov. 2015], 2015.

[76] Bundesamt für Sicherheit in der Informationstechnik. BSI TR-02102-1: Kryptographische Verfahren: Empfehlungen und Schlüssellängen (Version 2018-01). URL `https://www.bsi.bund.de/DE/Publikationen/TechnischeRichtlinien/tr02102/index_htm.html` [last accessed: Jan. 2018], 2018.

[77] Bundesfinanzhof (BFH). Formunwirksamkeit einer Klageerhebung per E-Mail ohne qualifizierte Signatur - Geringfügige Überschreitung der Frist zur Übermittlung der Revisionsbegründung - Wiedereinsetzung bei Fristversäumnis wegen Absturz der EDV-Anlage - "Unverzügliche Information" nach § 52a Abs. 2 Satz 3 FGO. Beschluss vom 26. 7. 2011 – VII R 30/10, 2011.

[78] Bundesgerichtshof (BGH). Urteil vom 11.7.1963 – VII ZR 120/62, 1963.

[79] Bundesgerichtshof (BGH). Urteil vom 29.02.1996 – IX ZR 153/95, 1996.

[80] Bundesgerichtshof (BGH). Urteil vom 21.12.2012 – VI ZB 28/10, 2012.

[81] Bundesgerichtshof (BGH). Beschluss vom 14. Mai 2013 – VI ZB 7/13, 2013.

[82] Bundesministerium des Inneren (BMI). Anwendungshinweise zum Informationsfreiheitsgesetz. Bekanntmachung des BMI vom 21. 11. 2005 - V 5a -130 250/16, 2005.

[83] Bundesministerium für Finanzen. Grundsätze zur ordnungsmäßigen Führung und Aufbewahrung von Büchern, Aufzeichnungen und Unterlagen in elektronischer Form sowie zum Datenzugriff (GoBD). IV A 4 - S 0316/13/10003, BStBl I S. 1450, 2014.

[84] Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen. Bekanntmachung zur elektronischen Signatur nach dem Signaturgesetz und der Signaturverordnung (Übersicht über geeignete Algorithmen) – Vom 15. 12. 2014. In *Bundesanzeiger BAnz AT 30.01.2015 B3*. Bundesanzeiger Verlag, 2015.

[85] Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen. Bekanntmachung zur elektronischen Signatur nach dem Signaturgesetz und der Signaturverordnung (Übersicht über geeignete Algorithmen) – Vom 7. Dezember 2016. In *BAnz AT 30.12.2016 B5*. Bundesanzeiger Verlag, 2016.

[86] Bundesnotarkammer. Rundschreiben 2006 - 25: Hinweise und Anwendungsempfehlungen für den elektronischen Handels-, Genossenschafts- und Partnerschaftsregisterverkehr, 2006. URL `http://www.bnotk.de/Bundesnotarkammer/Aufgaben-und-Taetigkeiten/Rundschreiben/2006-25.php` [last accessed: Sep. 2017].

[87] Bundesnotarkammer K.d.ö.R. Richtlinien für die Amtspflichten und sonstigen Pflichten der Mitglieder der Landesnotarkammer Bayern, 2007. URL `http://www.bnotk.de/_downloads/Richtlinien/Richtlinien-LNK-Bayern-neu.pdf` [last accessed: Sep. 2017].

[88] Bundestag und Bundesrat der Bundesrepublik Deutschland. Gesetz zur Regelung der Rahmenbedingungen für Informations- und Kommunikationsdienste Informations- und Kommunikationsdienste-Gesetz – IuKDG. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume 1, page 1870. Bundesanzeiger Verlag, 1997.

[89] Bundestag und Bundesrat der Bundesrepublik Deutschland. Criminal Code in the version promulgated on 13 November 1998, Federal Law Gazette [Bundesgesetzblatt] I p. 3322, last amended by Article 3 of the Law of 2 October 2009, Federal Law Gazette I p. 3214. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume 1, page 3214. Bundesanzeiger Verlag, 1998-2009. URL `http://www.gesetze-im-internet.de/englisch_stgb/german_criminal_code.pdf` [last accessed: Sep. 2017]. Translation provided by Prof. Dr. Michael Bohlander. The translation includes the amendment(s) to the Act by Article 3 of the Act of 2.10.2009 (Federal Law Gazette I p. 3214).

[90] Bundestag und Bundesrat der Bundesrepublik Deutschland. Umsatzsteuergesetz UStG (in der Fassung vom 21. Februar 2005). In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume 1, page 368. Bundesanzeiger Verlag, 2005.

[91] Bundestag und Bundesrat der Bundesrepublik Deutschland. Federal Data Protection Act (BDSG) in the version promulgated on 14 January 2003 (Federal Law Gazette I p. 66), as most recently amended by Article 1 of the Act of 14 August 2009 (Federal Law Gazette I p. 2814). In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, page 2005 page 3202; 2006 page 431; 2007 page 1781; 2009 page 3145. Bundesanzeiger Verlag, 2005-2009. URL `http://www.gesetze-im-internet.de/englisch_bdsg/englisch_bdsg.html` [last accessed: Sep. 2017]. Translations provided by the Language Service of the Federal Ministry of the Interior.

[92] Bundestag und Bundesrat der Bundesrepublik Deutschland. Federal Act Governing Access to Information held by the Federal Government (Freedom of Information Act) of 5 September 2005 (Federal Law Gazette [BGBl.] Part I, p. 2722), last amended by Article 2 (6) of the Act of 7 August 2013 (Federal Law Gazette I, p. 3154). In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume 1, page 3154. Bundesanzeiger Verlag, 2013.

[93] Bundestag und Bundesrat der Bundesrepublik Deutschland. Signaturgesetz vom 16. Mai 2001 (BGBl. I S. 876), das durch Artikel 4 Absatz 111 des Gesetzes vom 7. August 2013 (BGBl. I S. 3154) geaendert worden ist. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, pages 3154, Stand: Zuletzt geändert durch Art. 4 Abs. 111 G v. 7.8.2013 I 3154. Bundesanzeiger Verlag, 2013.

[94] Bundestag und Bundesrat der Bundesrepublik Deutschland. Signaturverordnung vom 16. November 2001 (BGBl. I S. 3074), die zuletzt durch Artikel 4 Absatz 112 des Gesetzes vom 7. August 2013 (BGBl. I S. 3154) geändert worden ist. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, pages 3154, Stand:Zuletzt geändert durch Art. 4 Abs. 112 G v. 7.8.2013 I 3154. Bundesanzeiger Verlag, 2013.

[95] Bundestag und Bundesrat der Bundesrepublik Deutschland. Gesetz zur Förderung des elektronischen Rechtsverkehrs. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume I, page 3786. Bundesanzeiger Verlag, 2013.

[96] Bundestag und Bundesrat der Bundesrepublik Deutschland. Zivilprozessordnung ZPO (Code of Civil Procedure) in der Fassung der Bekanntmachung vom 5. Dezember 2005 (BGBl. I S. 3202; 2006 I S. 431; 2007 I S. 1781), die zuletzt durch Artikel 6 des Gesetzes vom 20. November 2015 (BGBl. I S. 2018) geändert worden ist. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume 1, page 2018. Bundesanzeiger Verlag, 2015. URL `http://www.gesetze-im-internet.de/englisch_zpo/englisch_zpo.html` [last accessed: Sep. 2017]. A translation is provided by Samson-Übersetzungen GmbH, Dr. Carmen von Schöning but the translation includes the amendment(s) to the Act by Article 1 of the Act of 10.10.2013 (Federal Law Gazette I p. 3786).

[97] Bundestag und Bundesrat der Bundesrepublik Deutschland. De-Mail-Gesetz vom 28. April 2011 (BGBl. I S. 666), das durch Artikel 3 des Gesetzes vom 18. Juli 2017 (BGBl. I S. 2745) geändert worden ist. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, page 2745 Stand: Zuletzt geändert durch Art. 3 G v. 18.7.2017 I 2745. Bundesanzeiger Verlag, 2017.

[98] Bundestag und Bundesrat der Bundesrepublik Deutschland. Gesetz zur Durchführung der Verordnung (EU) Nr. 910/2014 des Europäischen Parlaments und des Rates vom 23. Juli 2014 über elektronische Identifizierung und Vertrauensdienste für elektronische Transaktionen im Binnenmarkt und zur Aufhebung der Richtlinie 1999/93/EG (eIDAS-Durchführungsgesetz). In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume 1, pages 2745–2756. Bundesanzeiger Verlag, 2017.

[99] Bundestag und Bundesrat der Bundesrepublik Deutschland. Zivilprozessordnung ZPO (Code of Civil Procedure) in der Fassung der Bekanntmachung vom 5. Dezember 2005 (BGBl. I S. 3202; 2006 I S. 431; 2007 I S. 1781), die zuletzt durch Artikel 11 Absatz 15 des Gesetzes vom 18. Juli 2017 (BGBl. I S. 2745) geändert worden ist. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*, volume 1, page 2745. Bundesanzeiger Verlag, 2017. URL `http://www.gesetze-im-internet.de/zpo/index.html` [last accessed: Sep. 2017].

[100] Bundesverfasssungsgericht (BVerfG). Judgment des Ersten Senats vom 27. February 2008. *1 BvR 370/07, 1 BvR 595/07 - NJW.* 2008. URL `http://www.bverfg.de/e/rs20080227_1bvr037007en.html` [last accessed: Sep. 2017].

[101] Bundesverwaltungsgericht (BVerwG). Urteil vom 23.06.2011 – Az. 20 F 21/10, 2011.

[102] D. Busch. Indirect Representation and the Lando Principles. *Electronic Journal of Comparative Law*, 2.3. 1998.

[103] P. Camacho and A. Hevia. Short transitive signatures for directed trees. In *Proc. of Topics in Cryptology – CT-RSA 2012*, volume 7178 of *LNCS*, pages 35–50. Springer, 2012. URL `https://doi.org/10.1007/978-3-642-27954-6_3` [last accessed: Sep. 2017].

[104] J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Proc. of Advances in Cryptology - CRYPTO 2002*, volume 2442 of *LNCS*, pages 61–76. Springer, 2002.

[105] J. Camenisch, M. Dubovitskaya, K. Haralambiev, and M. Kohlweiss. Composable & Modular Anonymous Credentials: Definitions and Practical Constructions. *IACR Cryptology ePrint Archive*, 580. 2015. URL `https://eprint.iacr.org/2015/580.pdf` [last accessed: Sep. 2017].

[106] J. Camenisch, M. Dubovitskaya, K. Haralambiev, and M. Kohlweiss. Composable and Modular Anonymous Credentials: Definitions and Practical Constructions. In *Proc. of Advances in Cryptology – ASIACRYPT 2015*, volume 9453 of *LNCS*, pages 262–288. Springer, 2015. URL `https://doi.org/10.1007/978-3-662-48800-3_11` [last accessed: Sep. 2017].

[107] J. Camenisch, D. Derler, S. Krenn, H. C. Pöhls, K. Samelin, and D. Slamanig. Chameleon-Hashes with Ephemeral Trapdoors And Applications to Invisible Sanitizable Signatures (full version). In *Proc. of Public-Key Cryptography (PKC 2017)*, volume 10175 of *LNCS*, pages 152–182. Springer, 2017. URL `https://eprint.iacr.org/2017/011` [last accessed: Sep. 2017]. Full Version: Cryptology ePrint Archive, Report 2017/11.

[108] M. Campagna, L. Chen, O. Dagdelen, J. Ding, J. K. Fernick, W. N. Gisin, D. Hayford, N. Lütkenhaus, M. Mosca, B. Neill, M. Pecen, G. Ribordy, J. M. Schanck, D. Stebila, N. Walenta, B. W. Whyte, Z. Zhang, S. Kaiser, A. Petzold, and D. Smith-Tone. *Quantum Safe Cryptography and Security – An introduction, benefits, enablers and challenges.* ETSI (European Telecommunications Standards Institute), 2015. ISBN 979-10-92620-03-0.

[109] S. Canard and A. Jambert. On Extended Sanitizable Signature Schemes. In *Proc. of Topics in Cryptology – CT-RSA 2010*, pages 179–194, 2010. URL `http://dx.doi.org/10.1007/978-3-642-11925-5_13` [last accessed: Sep. 2017].

[110] S. Canard and R. Lescuyer. Protecting privacy by sanitizing personal data: a new approach to anonymous credentials. In *Proc. of ACM Symposium on Information, Computer and Communications Security (ASIA CCS 2013)*, pages 381–392. ACM, 2013.

[111] S. Canard, F. Laguillaumie, and M. Milhau. Trapdoor Sanitizable Signatures and Their Application to Content Protection. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2008)*, volume 5037 of *LNCS*, pages 258–276. Springer, 2008. URL `http://dx.doi.org/10.1007/978-3-540-68914-0_16` [last accessed: Sep. 2017].

[112] S. Canard, A. Jambert, and R. Lescuyer. Sanitizable Signatures with Several Signers and Sanitizers. In *Progress in Cryptology – AFRICACRYPT 2012*, volume 7374 of *LNCS*, pages 35–52. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-31410-0_3` [last accessed: Sep. 2017].

[113] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. of IEEE Symposium on Foundations of Computer Science (FOCS 2001)*, pages 136–145. IEEE, 2001.

[114] R. Canetti, S. Halevi, and J. Katz. A Forward-Secure Public-Key Encryption Scheme. In *Proc. of Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 255–271. Springer, 2003. ISBN 3-540-14039-5. URL `http://dx.doi.org/10.1007/3-540-39200-9_16` [last accessed: Sep. 2017].

[115] V. T. Chakaravarthy, H. Gupta, P. Roy, and M. K. Mohania. Efficient techniques for document sanitization. In *Proc. of ACM conference on Information and knowledge management*, pages 843–852. ACM, 2008.

[116] E.-C. Chang and J. Xu. Remote Integrity Check with Dishonest Storage Server. In *Proc. of European Symposium on Research in Computer Security (ESORICS 2008)*, volume 5283 of *LNCS*, pages 223–237. Springer, 2008.

[117] E.-C. Chang, C. L. Lim, and J. Xu. Short Redactable Signatures Using Random Trees. In *Proc. of Topics in Cryptology – CT-RSA 2009*, volume 5473 of *LNCS*, pages 133–147. Springer, 2009. ISBN 978-3-642-00861-0. URL `http://dx.doi.org/10.1007/978-3-642-00862-7_9` [last accessed: Sep. 2017].

[118] S. Chari, T. Rabin, and R. L. Rivest. An Efficient Scheme for Route Aggregation. (unpublished), 2002. URL `https://people.csail.mit.edu/rivest/ChariRabinRivest-AnEfficientSignatureSchemeForRouteAggregation.pdf` [last accessed: Sep. 2017].

[119] M. Chase, M. Kohlweiss, A. Lysyanskaya, and S. Meiklejohn. Malleable Signatures: New Definitions and Delegatable Anonymous Credentials. In *2014 IEEE 27th Computer Security Foundations Symposium*, pages 199–213. IEEE, 2014.

[120] D. Chaum. Designated confirmer signatures. In *Proc. of Advances in Cryptology – EUROCRYPT '94*, volume 950 of *LNCS*, pages 86–91. Springer, 1995. ISBN 978-3-540-44717-7. URL `http://dx.doi.org/10.1007/BFb0053427` [last accessed: Sep. 2017].

[121] D. Chaum and E. van Heyst. Group Signatures. In *Proc. of Advances in Cryptology – EUROCRYPT '91*, volume 547 of *LNCS*, pages 257–265. Springer, 1991. URL `http://dx.doi.org/10.1007/3-540-46416-6_22` [last accessed: Sep. 2017].

[122] Z. Chen. *Java Card Technology for Smart Cards: Architecture and Programmer's Guide*. Addison-Wesley, 2000. ISBN 978-0201703290.

[123] T. Chik-How. On waters' signature scheme. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 89(10):2684–2685. IEICE, 2006.

[124] R. Chow, I. Oberst, and J. Staddon. Sanitization's slippery slope: The design and study of a text revision assistant. In *SOUPS 2009*, pages 13:1–13:11. ACM, 2009. ISBN 978-1-60558-736-3.

[125] D. D. Clark and D. R. Wilson. A Comparison of Commercial and Military Computer Security Policies. *IEEE Symposium on Security and Privacy*, 0:184. IEEE, 1987. ISSN 1540-7993.

[126] COMITÉ EUROPÉEN DE NORMALISATION (CEN). Guidelines for the implementation of secure signature-creation devices. CWA 14355:2004, 2004.

[127] J.-S. Coron and D. Naccache. Security analysis of the gennaro-halevi-rabin signature scheme. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 91–101. Springer, 2000.

[128] J.-S. Coron and D. Naccache. Boneh et al.'s k-Element Aggregate Extraction Assumption Is Equivalent to the Diffie-Hellman Assumption. In *Proc. of Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *LNCS*, pages 392–397. Springer, 2003.

[129] D. Crocker. STANDARD FOR THE FORMAT OF ARPA INTERNET TEXT MESSAGES. RFC 822, RFC Editor, 1982. URL `http://www.ietf.org/rfc/rfc822.txt` [last accessed: Sep. 2017]. Obsoleted by RFC 2822, updated by RFCs 1123, 2156, 1327, 1138, 1148.

[130] S. K. Das, R. H. Halverson, and K. B. Min. Efficient parallel algorithms for tree-related problems using the parentheses matching strategy. In *Proc. of 8th International Parallel Processing Symposium*, pages 362–367, 1994. doi: 10.1109/IPPS.1994.288276.

[131] H. de Meer, M. Liedel, H. C. Pöhls, J. Posegga, and K. Samelin. Indistinguishability of One-Way Accumulators (MIP-1210). Technical Report MIP-1210, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `https://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP_1210.pdf` [last accessed: Sep. 2017].

[132] H. de Meer, H. C. Pöhls, J. Posegga, and K. Samelin. Scope of Security Properties of Sanitizable Signatures Revisited. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2013)*, pages 188–197. IEEE, 2013. URL `http://dx.doi.org/10.1109/ARES.2013.26` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2013_DeMeer_Poehls_Posegga_Samelin-ScopeofSecurityPropertiesofSanitizableSignaturesRevisited.pdf` [last accessed: Sep. 2017].

[133] H. de Meer, H. C. Pöhls, J. Posegga, and K. Samelin. On the Relation between Redactable and Sanitizable Signature Schemes. In *Proc. of International Symposium on Engineering Secure Software and Systems (ESSoS 2014)*, volume 8364 of *LNCS*, pages 113–130. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-319-04897-0_8` [last accessed: Sep. 2017].

[134] H. de Meer, H. C. Pöhls, J. Posegga, and K. Samelin. Redactable Signature Schemes for Trees with Signer-Controlled Non-Leaf-Redactions. *Journal of E-Business and Telecommunications*, 455:155–171. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-662-44791-8_10` [last accessed: Sep. 2017].

[135] P. Decherney. *Hollywood's Copyright Wars: From Edison to the Internet*. Film and Culture Series. Columbia University Press, 2012. ISBN 9780231159463. URL `https://books.google.de/books?id=nsqrAgAAQBAJ` [last accessed: Sep. 2017].

[136] B. Deiseroth, V. Fehr, M. Fischlin, M. Maasz, N. F. Reimers, and R. Stein. Computing on Authenticated Data for Adjustable Predicates. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2013)*, volume 7954 of *LNCS*, pages 53–68. Springer, 2013.

[137] D. Demirel, D. Derler, C. Hanser, H. C. Pöhls, D. Slamanig, and G. Traverso. PRISMACLOUD public deliverable D4.4: Overview of Functional and Malleable Signature Schemes, 2016. URL `https://online.tugraz.at/tug_online/voe_main2.getvolltext?pCurrPk=86456` [last accessed: Sep. 2017].

[138] Der Bayerische Landesbeauftragte für den Datenschutz. Datenträgerentsorgung Orientierungshilfe. URL `https://www.datenschutz-bayern.de/technik/orient/oh_datentraegerentsorgung.pdf` [last accessed: Jan. 2017], 2014.

[139] D. Derler and D. Slamanig. Rethinking Privacy for Extended Sanitizable Signatures and a Black-Box Construction of Strongly Private Schemes. In *Proc. of International Conference on Provable Security (ProvSec 2015)*, volume 9451 of *LNCS*, pages 455–474. Springer, 2015. URL `https://doi.org/10.1007/978-3-319-26059-4_25` [last accessed: Sep. 2017].

[140] D. Derler, C. Hanser, and D. Slamanig. Privacy-Enhancing Proxy Signatures from Non-interactive Anonymous Credentials. In *Proc. of Data and Applications Security and Privacy (DBSec 2014)*, volume 8566 of *LNCS*, pages 49–65. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-662-43936-4_4` [last accessed: Sep. 2017].

[141] D. Derler, C. Hanser, and D. Slamanig. Revisiting Cryptographic Accumulators, Additional Properties and Relations to Other Primitives. In *Proc. of Topics in Cryptology – CT-RSA 2015*, volume 9048 of *LNCS*, pages 127–144. Springer, 2015. URL `http://dx.doi.org/10.1007/978-3-319-16715-2_7` [last accessed: Sep. 2017].

[142] D. Derler, C. Hanser, and D. Slamanig. Blank Digital Signatures: Optimization and Practical Experiences. In *Privacy and Identity Management for the Future Internet in the Age of Globalisation: 9th IFIP WG 9.2, 9.5, 9.6/11.7, 11.4, 11.6/SIG 9.2.2 International Summer School, Revised Selected Papers*, volume 457 of *IFIPAICT*, pages 201–215. Springer, 2015. URL `https://doi.org/10.1007/978-3-319-18621-4_14` [last accessed: Sep. 2017].

[143] D. Derler, H. C. Pöhls, K. Samelin, and D. Slamanig. A General Framework for Redactable Signatures and New Constructions. In *Proc. of International Conference on Information Security and Cryptology (ICISC 2015)*, volume 9558 of *LNCS*, pages 3–19. Springer, 2015. URL `https://link.springer.com/chapter/10.1007/978-3-319-30840-1_1` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2015_DerlerPoehlsSamelinSlamanig-GeneralFrameworkForRedactableSignatures_ICISC.pdf` [last accessed: Sep. 2017].

[144] D. Derler, C. Hanser, H. C. Pöhls, and D. Slamanig. Towards Authenticity and Privacy Preserving Accountable Workflows. In *Proc. of IFIP Summer School on Privacy and Identity Management*, volume 476 of *IFIPAICT*, pages 170–186. Springer, 2016. URL `https://doi.org/10.1007/978-3-319-41763-9_12` [last accessed: Sep. 2017]. `http://henrich.poehls.com/papers/2015_DerlerHanserPoehlsSlamanig_Towards_Auth_Private_Accountable_Workflows.pdf` [last accessed: Sep. 2017].

[145] D. Derler, S. Krenn, and D. Slamanig. Signer-Anonymous Designated-Verifier Redactable Signatures for Cloud-Based Data Sharing. In *Proc. of Cryptology and Network Security (CANS 2016)*, volume 10052 of *LNCS*, pages 211–227, 2016. ISBN 978-3-319-48964-3. URL `http://dx.doi.org/10.1007/978-3-319-48965-0_13` [last accessed: Sep. 2017].

[146] D. Derler, S. Krenn, and D. Slamanig. Signer-anonymous designated-verifier redactable signatures for cloud-based data sharing. In *Cryptology and Network Security*, pages 211–227, 2016. ISBN 978-3-319-48964-3.

[147] Deutscher Bundestag. Drucksache 14/4987, 2000. URL http://dip21.bundestag.de/dip21/btd/14/049/1404987.pdf [last accessed: Sep. 2017].

[148] Deutscher Bundestag. The Fiscal Code of Germany in the version promulgated on 1 October 2002. In *Bundesgesetzblatt (BGBl., Federal Law Gazette)*. Bundesanzeiger Verlag, 2002.

[149] Deutscher Bundestag. Drucksache 17/10720, 2012. URL http://dipbt.bundestag.de/dip21/btd/17/107/1710720.pdf [last accessed: Sep. 2017].

[150] Deutscher Bundestag. Gesetz zur Verbesserung der gesundheitsbezogenen Verbraucherinformation (Verbraucherinformationsgesetz - VIG). In *Federal Law Gazette [Bundesgesetzblatt]*. Bundesanzeiger Verlag, 2013.

[151] Deutscher Bundestag. Vertrauensdienstegesetz (VDG), 2017. URL http://dip21.bundestag.de/dip21/btd/18/124/1812494.pdf [last accessed: Sep. 2017].

[152] C.-E. Dietl and E. Lorenz. *Wörterbuch für Recht, Wirtschaft und Politik*. C.H.BECK, 6th edition, 2000. ISBN 978-3406441127.

[153] J. Dietrich and J. Keller-Herder. De-Mail – verschlüsselt, authentisch, nachweisbar. *Datenschutz und Datensicherheit - DuD*, 34(5):299–301. Springer, 2010. URL https://doi.org/10.1007/s11623-010-0091-6 [last accessed: Sep. 2017].

[154] W. Diffie and M. E. Hellman. New directions in cryptography. *Journal of Transactions on Information Theory*, 22(6):644–654. IEEE, 1976.

[155] DIN. DIN 66399-2: Büro- und Datentechnik – Vernichten von Datenträgern – Teil 2: Anforderungen an Maschinen zur Vernichtung von Datenträgern, 2012.

[156] R. H. Dolin, L. Alschuler, C. Beebe, P. V. Biron, S. L. Boyer, D. Essin, E. Kimber, T. Lincoln, and J. E. Mattison. The HL7 Clinical Document Architecture. *Journal of the American Medical Informatics Association*, 8(6):552–569. Oxford University Press, 2001. URL http://dx.doi.org/10.1136/jamia.2001.0080552 [last accessed: Sep. 2017].

[157] J. Dumortier. Legal Status of Qualified Electronic Signatures in Europe. In *Proc. of ISSE 2004 – Securing Electronic Business Processes*, pages 281–289. Vieweg, 2004. URL https://doi.org/10.1007/978-3-322-84984-7_28 [last accessed: Sep. 2017].

[158] P. Dusart. Estimates of some functions over primes without R.H. *Arxiv preprint arXiv*, 1002.0442. 2010. URL https://arxiv.org/pdf/1002.0442 [last accessed: Sep. 2017].

[159] C. Dwork. Differential Privacy. In *Proc. of International Conference on Automata, Languages and Programming (ICALP '06)*, volume 4052 of *LNCS*, pages 1–12. Springer, 2006. URL http://dx.doi.org/10.1007/11787006_1 [last accessed: Sep. 2017].

[160] T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proc. of Advances in Cryptology – CRYPTO '85*, volume 196 of *LNCS*, pages 10–18. Springer, 1985. URL https://doi.org/10.1007/3-540-39568-7_2 [last accessed: Sep. 2017].

[161] Electronic Frontier Foundation (EFF). Picture from 'EFF Victory Results in Release of Secret Court Opinion Finding NSA Surveillance Unconstitutional'. URL https://www.eff.org/files/images_insert/fisc_5.png [last accessed: Jun. 2017], 2013.

[162] C. Ellison and B. Schneier. Ten risks of PKI: What you're not being told about public key infrastructure. *Computer Security Journal*, 16(1):1–7. 2000. URL https://www.schneier.com/academic/paperfiles/paper-pki.pdf [last accessed: Sep. 2017].

[163] C. M. Ellison. Non-repudiation. URL http://www.std.com/~cme/non-repudiation.htm [last accessed: Jan. 2016], not stated.

[164] M. Enev, S. Gupta, T. Kohno, and S. N. Patel. Televisions, video privacy, and powerline electromagnetic interference. In *Proc. of ACM SIGSAC Symposium on Information, Computer and Communications Security (ASIA CCS 2011)*, pages 537–550. ACM, 2011. ISBN 978-1-4503-0948-6.

[165] A. Erwig, M. Fischlin, M. Hald, D. Hehm, R. Kiel, F. Kübler, M. Kümmerlin, J. Laenge, and F. Rohrbach. Redactable Graph Hashing, Revisited. In *Proc. of Australasian Conference on Information Security and Privacy (ACISP 2017)*, volume 10342 of *LNCS*, pages 398–405. Springer, 2017. URL https://doi.org/10.1007/978-3-319-59870-3_24 [last accessed: Sep. 2017].

[166] EU Article 29 Data Protection Working Party. Working Document 1/2009 on pre-trial discovery for cross border civil litigation. 00339/09/EN, WP 158 URL `http://henrich.poehls.com/papers/wp158_en.pdf` [last accessed: Jan. 2018], 2009. URL `http://www.gpdp.gov.mo/uploadfile/others/wp158_en.pdf` [last accessed: Sep. 2017].

[167] EU Court of Justice. Judgement of the court Case C, 28/8 P, 2010.

[168] EU Court of Justice. Judgement of the court Case C, 131/12. URL `http://curia.europa.eu/juris/document/document_print.jsf?doclang=EN&docid=152065` [last accessed: Jan. 2017], 2014.

[169] European Commission. Electronic identification, signatures and trust services: Questions and Answers. URL `http://europa.eu/rapid/press-release_MEMO-12-403_en.htm?locale=en`, 2012.

[170] European Commission. European Commission: List of Trusted List information as notified by Member States. URL `http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?doc_id=1788` [last accessed: Apr. 2016], 2016.

[171] European Commission. Compilation of: Member States' notifications on: Designated Bodies under Article 30(2) and 39(2) of Regulation 910/2014 and Certified Qualified Signature Creation Devices under Article 31(1)-(2),..., 2017. URL `https://ec.europa.eu/futurium/en/system/files/ged/sscds_qscds_list2017-10-27.pdf` [last accessed: Sep. 2017].

[172] European Commission DG Internal Market and Financing Services (DG XV). *Digital Signatures: A survey of law and practise in the European Union*. Woodhead Publishing Limited, 2000. ISBN 1855734699.

[173] European Commission Directorate-General for Translation. *English Style Guide — A handbook for authors and translators in the European Commission*. European Commission Directorate-General Translation, 7th edition, 2014. URL `http://ec.europa.eu/translation/english/guidelines/documents/styleguide_english_dgt_en.pdf` [last accessed: Sep. 2017].

[174] European Parliament and the Council of the European Union. Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. *Official Journal*, OJ L 281 of 23.11.1995:31 – 50. 1995.

[175] European Parliament and the Council of the European Union. Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures. *Official Journal*, L 013 of 19/01/2000:12–20. 2000.

[176] European Parliament and the Council of the European Union. Regulation 45/2001 of the European Parliament and of the Council of 18 December 2000 on the protection of individuals with regard to the processing of personal data by the Community institutions and bodies and on the free movement of such data. *Official Journal*, OJ L 8 of 12.1.2001:1–22. 2001.

[177] European Parliament and the Council of the European Union. Regulation (EC) No 1049/2001 of the European Parliament and of the Council of 30 May 2001 regarding public access to European Parliament, Council and Commission documents. *Official Journal*, OJ L 145 of 31.5.2001. 2001.

[178] European Parliament and the Council of the European Union. Regulation (EC) No 178/2002 of the European Parliament and of the Council of 28 January 2002 laying down the general principles and requirements of food law, establishing the European Food Safety Authority and laying down procedures in matters of food safety. *Official Journal*, OJ L 031 of 1.2.2002:1–24. 2002.

[179] European Parliament and the Council of the European Union. Regulation 460/2004/EC of the European Parliament and of the Council of Europe of 10 March 2004 establishing the European Network and Information Security Agency. *Official Journal*, OJ L 077 of 13/03/2004:1–11. 2004.

[180] European Parliament and the Council of the European Union. Directive 2009/140/EC of 25 November 2009 amending Directives 2002/21/EC on a common regulatory framework for electronic communications networks and services, 2002/19/EC on access to, and interconnection of, electronic communications networks and associated facilities, and 2002/20/EC on the authorization of electronic communications networks and services. *Official Journal*, OJ L 337/8 of 2009/12/18. 2009.

[181] European Parliament and the Council of the European Union. Commission Implementing Regulation No 931/2011 on the traceability requirements for food of animal origin set by Regulation (EC) No 178/2002 for food of animal origin. *Official Journal*, OJ L 242/2. 2011.

[182] European Parliament and the Council of the European Union. Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC. *Official Journal*, OJ L 257 of 28.8.2014:73–114. 2014.

[183] European Parliament and the Council of the European Union. Directive (EU) 2016/943 of the European Parliament and of the Council of 8 June 2016 on the protection of undisclosed know-how and business information (trade secrets) against their unlawful acquisition, use and disclosure. *Official Journal*, L 157: 1–18. 2016.

[184] European Parliament and the Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). *Official Journal*, OJ L 119 of 4.5.2016:1–88. 2016.

[185] European Smart Grids Task Force – Expert Group 1 – Standards and Interoperability for Smart Grids Deployment. My Energy Data. URL `https://ec.europa.eu/energy/sites/ener/files/documents/report_final_eg1_my_energy_data_15_november_2016.pdf` [last accessed: Jul. 2017], 2016.

[186] European Union Agency for Network and Information Security (ENISA). Shortlisting network and information security standards and good practices – Version 1.0, 2012. URL `https://resilience.enisa.europa.eu/article-13/shortlist-of-networks-and-information-security-standards` [last accessed: Sep. 2017].

[187] European Union Agency for Network and Information Security (ENISA). Recommended cryptographic measures - securing personal data, 2013. URL `https://www.enisa.europa.eu/publications/recommended-cryptographic-measures-securing-personal-data/at_download/fullReport` [last accessed: Sep. 2017].

[188] European Union Agency for Network and Information Security (ENISA). Algorithms, key size and parameters report 2014, 2014. URL `http://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-size-and-parameters-report-2014` [last accessed: Sep. 2017].

[189] Federal Ministry of Food and Agriculture. Das Verbraucher-Informationsgesetz (VIG). URL `http://www.bmel.de/DE/Ministerium/_Texte/Verbraucherinformationsgesetz.html` [last accessed: Jan. 2017], 2012.

[190] H. Federrath, M. Hansen, and M. Waidner. Andreas pfitzmann 1958-2010: Pioneer of technical privacy protection in the information society. In *Privacy and Identity Management for Life*, volume 352 of *IFIPAICT*, pages 349–352. Springer, 2011.

[191] V. Fehr and M. Fischlin. Sanitizable Signcryption: Sanitization over Encrypted Data (full version). *IACR Cryptology ePrint Archive*, 765. 2015. URL `http://eprint.iacr.org/2015/765.pdf` [last accessed: Sep. 2017].

[192] A. Fiat and A. Shamir. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Advances in Cryptology — CRYPTO' 86*, volume 263 of *LNCS*, pages 186–194. Springer, 1987.

[193] S. Fischer-Dieskau, R. Gitter, S. Paul, and R. Steidle. Elektronisch signierte Dokumente als Beweismittel im Zivilprozess. *MMR - MultiMedia und Recht*, page 709. C.H.BECK, 2002.

[194] S. Fischer-Dieskau, R. Gitter, and G. Hornung. Die Beschränkung des qualifizierten Zertifikats. *MMR - MultiMedia und Recht*, page 384. C.H.BECK, 2003.

[195] S. Fischer-Dieskau, A. Roßnagel, and R. Steidle. Beweisführung am seidenen Bit-String? - Die Langzeitaufbewahrung elektronischer Signaturen auf dem Prüfstand. *MMR - MultiMedia und Recht*, page 451. C.H.BECK, 2004.

[196] N. Fleischhacker, J. Krupp, G. Malavolta, J. Schneider, D. Schröder, and M. Simkin. Efficient unlinkable sanitizable signatures from signatures with rerandomizable keys. *IACR Cryptology ePrint Archive*, 395. 2015.

[197] N. Fleischhacker, J. Krupp, G. Malavolta, J. Schneider, D. Schröder, and M. Simkin. Efficient unlinkable sanitizable signatures from signatures with re-randomizable keys. In *Proc. of Public-Key Cryptography (PKC 2016)*, volume 9614 of *LNCS*, pages 301–330. Springer, 2016. ISBN 978-3-662-49384-7. URL `http://dx.doi.org/10.1007/978-3-662-49384-7_12` [last accessed: Sep. 2017].

[198] C. Frädrich, H. C. Pöhls, W. Popp, N. Rakotondravony, and K. Samelin. Integrity and Authenticity Protection with Selective Disclosure Control in the Cloud & IoT. In *Proc. of Information and Communications Security (ICICS 2016)*, volume 9977 of *LNCS*, pages 197–213. Springer, 2016. URL http://henrich.poehls.com/papers/2016_SelectiveDisclosureControl_ICICS2016_full.pdf [last accessed: Sep. 2017]. URL https://doi.org/10.1007/978-3-319-50011-9_16 [last accessed: Sep. 2017].

[199] F.W.J. van Geelkerken, H. C. Pöhls, and S. Fischer-Hübner. The legal status of malleable- and functional signatures in light of Regulation (EU) No 910/2014. In *Proc. of International Academic Conference of Young Scientists on Law & Psychology 2015 (LPS 2015)*, pages 404–410. L'viv Polytechnic Publishing House, 2015. ISBN 978-617-607-856-2. URL https://drive.google.com/file/d/0B-Yu3Ni9z3PXM2lBajhCXzhoWk0/view [last accessed: Sep. 2017]. URL http://henrich.poehls.com/papers/2015_GeelkerkenPoehlsHuebner_legal_status_of_malleable_and_functional_signatures_in_eidas.pdf [last accessed: Sep. 2017].

[200] S. Gajek, L. Liao, and J. Schwenk. Breaking and fixing the inline approach. In *Proc. of Workshop on Secure Web Services (SWS 2007)*, pages 37–43. ACM, 2007. URL http://doi.acm.org/10.1145/1314418.1314425 [last accessed: Sep. 2017].

[201] S. Gajek, J. Seedorf, and O. Dagdelen. Method and system for modifying an authenticated and/or encrypted message. URL https://www.google.com/patents/EP2719149B1 [last accessed: Jan. 2017], 2015. EU Patent – EP 2719149.

[202] R. Gennaro, S. Halevi, and R. Rabin. Secure Hash-and-Sign Signatures Without the Random Oracle. In *Proc. of Advances in Cryptology – EUROCRYPT 1999*, volume 1592 of *LNCS*, pages 123–139. Springer, 1999.

[203] K. Gennen. Praktischer Einsatz elektronischer Signaturen in Deutschland. *Datenschutz und Datensicherheit - DuD*, 33(11):661–664. Springer, 2009. ISSN 1862-2607. URL http://dx.doi.org/10.1007/s11623-009-0175-3 [last accessed: Sep. 2017].

[204] C. B. Gentry, Z. Ramzan, and B. Bruhn. Redactable Signatures for Certificates. URL http://www.google.com.na/patents/EP1843514A2 [last accessed: Sep. 2017], 2005. EU Patent – EP 1843514 A2.

[205] Germany. Bürgerliches Gesetzbuch (BGB) / German Civil Code, The translation includes the amendment(s) to the Act by Article 4 para. 5 of the Act of 1.10.2013 (Federal Law Gazette I p. 3719). It includes those amendments that entered into force on 1.1.2014. URL http://www.gesetze-im-internet.de/englisch_bgb/german_civil_code.pdf [last accessed: Sep. 2017].

[206] G. . D. GmbH. SmartC@fé® Expert 4.0 V.05.2008, 2008.

[207] O. Goldreich. On Post-Modern Cryptography. URL http://www.wisdom.weizmann.ac.il/~oded/PDF/on-pmc1.pdf [last accessed: Aug. 2016], 2006.

[208] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge university press, 2009. ISBN 978-0521119917.

[209] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4): 792–807. ACM, 1986. URL http://doi.acm.org/10.1145/6490.6503 [last accessed: Sep. 2017].

[210] S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2): 270–299. Elsevier, 1984.

[211] S. Goldwasser, S. Micali, and R. L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM Journal on Computing*, 17:281–308. SIAM, 1988.

[212] D. Gollmann. *Computer Security*. John Wiley & Sons, 1st edition, 1999. ISBN 0471978442.

[213] D. Gollmann. *Computer Security*. John Wiley & Sons, 2nd edition, 2006. ISBN 0470862939.

[214] D. Gollmann. *Computer Security*. John Wiley & Sons, 3rd edition, 2011. ISBN 0470741153.

[215] D. Gollmann. Veracity, plausibility, and reputation. In *Proc. of IFIP Workshop on Information Security Theory and Practice (WISTP 2012)*, volume 7322 of *LNCS*, pages 20–28. Springer, 2012.

[216] J. Gong, H. Qian, and Y. Zhou. Fully-Secure and Practical Sanitizable Signatures. In *Proc. of International Conference on Information Security and Cryptology (INSCRYPT 2010)*, volume 6584 of *LNCS*, pages 300–317. Springer, 2011. URL https://doi.org/10.1007/978-3-642-21518-6_21 [last accessed: Sep. 2017].

[217] V. Goyal, A. ONeill, and V. Rao. Correlated-input secure hash functions. In *Theory of Cryptography*, pages 182–200. Springer, 2011.

[218] U. Greveler, B. Justus, and D. Löhr. Identifikation von Videoinhalten über granulare Stromverbrauchsdaten. In *Proc. of the Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik (GI Sicherheit 2012)*, volume 195 of *GI-Edition Lecture Notes in Informatics (LNI)*, pages 35–45. GI, 2012. ISBN 978-3-88579-289-5.

[219] O. Grigorjew. *Beweiseignung fortgeschrittener elektronischer Signaturen*. Kassel University, 2014. ISBN 978-3862199600.

[220] P. Gutmann. PKI: Its Not Dead, Just Resting. *IEEE Computer*, pages 41–49. IEEE, 2002.

[221] H. C. Pöhls et al. RERUM Deliverable D3.2 - Privacy enhancing techniques in the Smart City applications, 2015. URL `http://cordis.europa.eu/docs/projects/cnect/4/609094/080/deliverables/001-RERUMdeliverableD32Ares20153669911.pdf` [last accessed: Sep. 2017].

[222] S. Haber, Y. Hatano, Y. Honda, W. G. Horne, K. Miyazaki, T. Sander, S. Tezoku, and D. Yao. Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In *Proc. of the ACM Symposium on Information, Computer and Communications Security (ASIA CCS 2008)*, pages 353–362. ACM, 2008. URL `http://doi.acm.org/10.1145/1368310.1368362` [last accessed: Sep. 2017].

[223] S. Haber, W. Horne, and M. Zhang. Redactable document signatures. URL `http://www.google.com/patents/WO2015156786A1` [last accessed: Sep. 2017], 2015. WO Patent App. PCT/US2014/033,366.

[224] C. Hanser and D. Slamanig. Blank Digital Signatures (full version). In *Proc. of ACM Symposium on Information, Computer and Communications Security (ASIA CCS 2013)*, pages 95–106. ACM, 2013. Full Version: Cryptology ePrint Archive, Report 2013/130.

[225] R. Herbst. *Wörterbuch der Handels-, Finanz-und Rechtssprache*. Springer, 2012. ISBN 1461261236.

[226] R. Herkenhöner, H. de Meer, M. Jensen, and H. C. Pöhls. Towards Automated Processing of the Right of Access in Inter-organizational Web Service Compositions. In *Proc. of IEEE World Congress on Services (SERVICES 2010)*, pages 645–652. IEEE Computer Society, 2010. ISBN 978-0-7695-4129-7. URL `http://dx.doi.org/10.1109/SERVICES.2010.56` [last accessed: Sep. 2017].

[227] P. D. Hert and S. Gutwirth. Privacy, data protection and law enforcement. opacity of the individual and transparency of power. In *Privacy and the criminal law*, pages 61–104. Intersentia nv, 2006. ISBN 9789050955454.

[228] Hessisches VGH. Beschluss vom 02.08.2012 – Az. 27 F 96/11. URL `http://openjur.de/u/441533.html` [last accessed: Jan. 2016], 2012.

[229] A. Hevia and D. Micciancio. The provable security of graph-based one-time signatures and extensions to algebraic signature schemes. In *Proc. of Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 379–396. Springer, 2002. URL `https://doi.org/10.1007/3-540-36178-2_24` [last accessed: Sep. 2017].

[230] Hitachi, Ltd. Glossary entry: Redactable signatures. URL `http://wayback.archive.org/web/20130106021344/http://www.hitachi.com/rd/yrl/glossary/r/redactable_signature.html` [last accessed Jan. 2016], 2016.

[231] F. Höhne and H. C. Pöhls. Staatliche Schutzpflichten für die IT-Infrastruktur. In P. Schartner and E. Weippl, editors, *Proc. of D-A-CH Security 2010*. syssec, 2010. ISBN 978-3-00-031441-4.

[232] F. Höhne and H. C. Pöhls. Grund und Grenzen staatlicher Schutzpflichten für die IT-Infrastruktur. In J. Taeger, editor, *Tagungsband der 11. Herbstakademie der Deutschen Stiftung für Recht und Informatik (DSRI): Digitale Evolution - Herausforderungen für das Informations- und Medienrecht*. OlWIR Oldenburger Verlag für Wirtschaft, Informatik und Recht, 2010. ISBN 978-3-939704-50-8.

[233] F. Höhne, H. C. Pöhls, and K. Samelin. Rechtsfolgen editierbarer Signaturen. *Datenschutz und Datensicherheit - DuD*, 36(7):485–491. Springer, 2012. URL `http://dx.doi.org/10.1007/s11623-012-0165-8` [last accessed: Sep. 2017].

[234] G. Hornung. Ein neues Grundrecht. *CR — Computer und Recht*, 5:299–306. Verlag Dr. Otto Schmidt Köln, 2008. URL `https://www.uni-kassel.de/fb07/fileadmin/datas/fb07/5-Institute/IWR/Hornung/Hornung__Ein_neues_Grundrecht__CR_2008__299.pdf` [last accessed: Sep. 2017].

[235] G. Hornung. Eine Datenschutz-Grundverordnung für Europa? — Licht und Schatten im Kommissionsentwurf vom 25.1.2012. *Zeitschrift für Datenschutz - ZD*, 25(3):99–106. C.H.BECK, 2012.

[236] R. Housley, W. Polk, W. Ford, and D. Solo. Internet X.509 PKI Certificate and Certificate Revocation List (CRL) Profile. RFC 3280, RFC Editor, 2002. URL `http://www.ietf.org/rfc/rfc3280.txt` [last accessed: Sep. 2017].

[237] G. A. Huff. Trusted Computer Systems – Glossary. Technical Report MTR-8201, MITRE, 1981. URL `www.dtic.mil/get-tr-doc/pdf?AD=ADA108829` [last accessed: Sep. 2017].

[238] T. B. Idalino, L. Moura, R. F. Custódio, and D. Panario. Locating modifications in signed data for partial data integrity. *Information Processing Letters*. Elsevier, 2015. URL `http://dx.doi.org/10.1016/j.ipl.2015.02.014` [last accessed: Sep. 2017].

[239] ISO/IEC. ISO/IEC 7498-2: Information processing systems — Open Systems Interconnection — Basic Reference Model — Part 2: Security Architecture. *ISO Geneve, Switzerland.* 1989.

[240] ISO/IEC. ISO/IEC 10118-1: Information technology — Security techniques — Hash-functions. *ISO Geneve, Switzerland.* 1996.

[241] ISO/IEC. ISO/IEC 10181-4: Information technology — Open Systems Interconnection — Security frameworks for open systems: Non-repudiation framework. *ISO Geneve, Switzerland.* 1996.

[242] ISO/IEC. ISO/IEC 9798-1: Information technology — Security techniques — Entity authentication — Part1: General. *ISO Geneve, Switzerland.* 1996.

[243] ISO/IEC. ISO/IEC 13888-1: Information technology — Security techniques — Non-repudiation — Part 1: General. *ISO Geneve, Switzerland.* 1997.

[244] ISO/IEC. ISO/IEC TR 14516: Information technology — Security techniques — Guidelines for the use and management of Trusted Third Party services. *ISO Geneve, Switzerland.* 2002.

[245] ISO/IEC. ISO/IEC 27000: Information technology — Security techniques — Information security management systems — Overview and vocabulary. *ISO Geneve, Switzerland.* 2005.

[246] ISO/IEC. ISO/IEC 27001: Information technology — Security techniques — Information security management systems — Requirements. *ISO Geneve, Switzerland.* 2005.

[247] ISO/IEC. ISO-IEC 11770-3: Information technology — Security techniques — Key management — Part 3: Mechanisms using asymmetric techniques. *ISO Geneve, Switzerland.* 2008.

[248] ISO/IEC. ITU-T Recommendation X.509 | ISO/IEC 9594-8: Information Technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks. *ISO Geneve, Switzerland.* 2008.

[249] ISO/IEC. ISO/IEC 27000: Information technology — Security techniques — Information security management systems — Overview and vocabulary. *ISO Geneve, Switzerland.* 2009.

[250] ISO/IEC. ISO/IEC 29100:2011: Information technology — Security techniques — Privacy framework. *ISO Geneve, Switzerland.* 2011.

[251] ISO/IEC. ISO/IEC 27037: Information technology — Security techniques — Guidelines for identification, collection, acquisition and preservation of digital evidence. *ISO Geneve, Switzerland.* 2012.

[252] ISO/IEC. ISO/IEC 27001: Information technology — Security techniques — Information security management systems — Requirements. *ISO Geneve, Switzerland*, 2nd edition. 2013.

[253] ISO/IEC. ISO-IEC 17788: Information technology — Cloud computing — Overview and vocabulary. *ISO Geneve, Switzerland.* 2014.

[254] ISO/IEC. ISO/IEC 27000: Information technology — Security techniques — Information security management systems — Overview and vocabulary. *ISO Geneve, Switzerland*, 3rd edition. 2014.

[255] ISO/IEC. ISO/IEC 27018: Information technology — Security techniques — Code of practice for protection of personally identifiable information (PII) in public clouds acting as PII processors. *ISO Geneve, Switzerland*, 4t edition. 2014.

[256] ISO/IEC. ISO/IEC 27038:2014: Information technology — Security techniques — Specification for digital redaction. *ISO Geneve, Switzerland.* 2014.

[257] ISO/IEC. ISO/IEC 27040: Information technology — Security techniques — Storage security. *ISO Geneve, Switzerland.* 2015.

[258] ISO/IEC. ISO/IEC 27042: Information technology — Security techniques — Guidelines for the analysis and interpretation of digital evidence. *ISO Geneve, Switzerland.* 2015.

[259] ISO/IEC. ISO/IEC 2832: Information technology – Vocabulary. *ISO Geneve, Switzerland.* 2015.

[260] ISO/IEC. ISO/IEC 27000: Information technology — Security techniques — Information security management systems — Overview and vocabulary. *ISO Geneve, Switzerland*, 4th edition. 2016.

[261] ISO/IEC. ISO/IEC 27050-1: Information technology — Security techniques — Electronic discovery — Part 1: Overview and concepts. *ISO Geneve, Switzerland.* 2016.

[262] G. Itkis. *Handbook of Information Security*, chapter Forward security, adaptive cryptography: Time evolution. John Wiley and Sons, 2006.

[263] ITU. *Security architecture for Open Systems Interconnection for CCITT applications.* International Telecommunications Union, 1991.

[264] T. Izu, N. Kanaya, M. Takenaka, and T. Yoshioka. PIATS: A Partially Sanitizable Signature Scheme. In *7th International Conference Information and Communications Security (ICICS)*, volume 3783 of *LNCS*, pages 72–83. Springer, 2005. URL `http://dx.doi.org/10.1007/11602897_7` [last accessed: Sep. 2017].

[265] T. Izu, N. Kunihiro, K. Ohta, M. Sano, and M. Takenaka. Information security applications. In *Proc. of International Workshop on Information Security Applications (WISA 2008)*, chapter Sanitizable and Deletable Signature, pages 130–144. Springer, 2009. ISBN 978-3-642-00305-9. URL `http://dx.doi.org/10.1007/978-3-642-00306-6_10` [last accessed: Sep. 2017].

[266] T. Izu, M. Izumi, N. Kunihiro, and K. Ohta. Yet another sanitizable and deletable signatures. In *AINA Workshops*, pages 574–579, 2011.

[267] T. Izu, M. Takenaka, J. Yajima, and T. Yoshioka. Integrity Assurance for Real-Time Video Recording. In *Proc. of Int. Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS 2012)*, pages 651–655. IEEE, IEEE, 2012.

[268] T. Jager, S. Schinzel, and J. Somorovsky. Bleichenbacher's Attack Strikes again: Breaking PKCS#1 v1.5 in XML Encryption. In *Proc. of European Symposium on Research in Computer Security (ESORICS 2012)*, volume 7459 of *LNCS*, pages 752–769. Springer, 2012. URL `https://doi.org/10.1007/978-3-642-33167-1_43` [last accessed: Sep. 2017].

[269] M. Jakobsson, K. Sako, and R. Impagliazzo. Designated verifier proofs and their applications. In *Proc. of Advances in Cryptology – EUROCRYPT '96*, volume 1070 of *LNCS*, pages 143–154. Springer, 1996. URL `https://doi.org/10.1007/3-540-68339-9_13` [last accessed: Sep. 2017].

[270] S. Jandt. Beweissicherheit im elektronischen Rechtsverkehr. *NJW Neue juristische Wochenzeitung*, page 1205. C.H.BECK, 2015.

[271] S. Jandt, T. Michalek, and K. Dietrich. Wie hoch ist der (Beweis-) Wert digitaler Dokumente? *Datenschutz und Datensicherheit - DuD*, 39(10):687–691. Springer, 2015. URL `http://dx.doi.org/10.1007/s11623-015-0499-0` [last accessed: Sep. 2017].

[272] M. Jawurek. *Privacy in Smart Grids*. PhD thesis, Friedrich-Alexander-University Erlangen-Nuernberg, 2013.

[273] R. Johnson, D. Molnar, D. Song, and D. Wagner. Homomorphic signature schemes. In *Proc. of the RSA Security Conference - Cryptographers Track*, pages 244–262. Springer, 2002.

[274] J. Jonsson and B. Kaliski. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. RFC 3447, RFC Editor, 2003. URL `http://www.ietf.org/rfc/rfc3447.txt` [last accessed: Sep. 2017].

[275] H. Joo. Content authentication scheme for modifiable multimedia streams. In *Proc. of International Conference on Multimedia, Computer Graphics and Broadcasting (MulGraB 2011)*, volume 262 of *CCIS*, pages 377–386. Springer, 2011. URL `http://dx.doi.org/10.1007/978-3-642-27204-2_45` [last accessed: Sep. 2017].

[276] A. Jøsang and B. Alfayyadh. Robust WYSIWYS: a method for ensuring that what you see is what you sign. In *Proc. of Australasian Information Security Conference (AISC 2008)*, volume 30, pages 53–58. Australian Computer Society, Inc., 2008. URL `http://urn.nb.no/URN:NBN:no-21754` [last accessed: Sep. 2017].

[277] S. Jungermann. *Der Beweiswert elektronischer Signaturen*, volume 9 of *Schriften zum Handels- und Wirtschaftsrecht*. Lang, 2002. ISBN 3-631-50145-5.

[278] J. Katz and Y. Lindell. *Introduction to modern cryptography*. CRC Press, 2014. ISBN 978-1466570269.

[279] S. Katzenbeisser and F. Petitcolas. *Information hiding techniques for steganography and digital watermarking*. Artech house, 1999. ISBN 978-1580530354.

[280] W. Kilian. EG-Richtlinie über digitale Signaturen in Kraft. *Betriebs-Berater (BB): Recht, Wirtschaft, Steuern*, 15:733–735. Deutscher Fachverlag, 2000.

[281] E. Kiltz, A. Mityagin, S. Panjwani, and B. Raghavan. Append-only signatures. In *Proc. of 32nd International Colloquium Automata, Languages and Programming (ICALP 2005)*, volume 3580 of *LNCS*, pages 434–445. Springer, 2005. URL `http://dx.doi.org/10.1007/11523468_36` [last accessed: Sep. 2017].

[282] S. Kiran, P. Lareau, and S. Lloyd. PKI Basics - A Technical Perspective, 2002. URL `www.oasis-pki.org/pdfs/PKI_Basics-A_technical_perspective.pdf` [last accessed: Sep. 2017]. Deliverable from the PKI Forum's Business Working Group (BWG).

[283] R. Kissel, A. Regenscheid, M. Scholl, and K. Stine. NIST Special Publication 800-88 – Revision 1 – Guidelines for Media Sanitization. National Institute of Standards and Technology (NIST), `http://dx.doi.org/10.6028/NIST.SP.800-88r1` [last accessed: Jan 2017], 2014.

[284] S. Klein. Die Beweiskraft elektronischer Verträge Zur Entwicklung der zivilprozessrechtlichen Vorschriften über die Beweiskraft elektronischer Dokumente. *JurPC: Internet-Zeitschrift für Rechtsinformatik*, 198. 2007.

[285] M. Klonowski and A. Lauks. Extended Sanitizable Signatures. In *Proc. of International Conference on Information Security and Cryptology (ICISC 2006)*, volume 4296 of *LNCS*, pages 343–355. Springer, 2006. URL `https://link.springer.com/chapter/10.1007/11927587_28` [last accessed: Sep. 2017].

[286] V. Kment. European regulation eIDAS: The impuls to unify the electronic signature and identification in the EU. *Jurisprudence*, Season XXIII(6):25–35. Wolters Kluwer, 2014.

[287] N. Koblitz and A. J. Menezes. Another Look at "Provable Security". *Journal of Cryptology*, 20(3). Springer, 2007.

[288] T. Kono, C. Paulus, and H. Rajak, editors. *Selected Legal issues Issues of E-commerce*. Kluwer Law Internationl, 2002. ISBN 9041118985.

[289] Koordinierungsstelle für IT-Standards (KoSIT). OSCI-Transport (OSCI 1.2 / OSCI 2) und XTA. URL `http://www.xoev.de/die_standards/osci_transport-3355` [last accessed: Jan. 2017], 2017.

[290] E. Kosta and K. Stuurman. Technical Standards and the Draft General Data Protection Regulation. In *The Law, Economics and Politics of International Standardization*, pages 394–433. Cambridge University Press, 2016. URL `https://doi.org/10.1017/CBO9781316423240` [last accessed: Sep. 2017].

[291] H. Krawczyk and T. Rabin. Chameleon Hashing and Signatures. *IACR Cryptology ePrint Archive*, 010. 1998. URL `https://eprint.iacr.org/1998/010.ps` [last accessed: Sep. 2017].

[292] H. Krawczyk and T. Rabin. Chameleon Hashing and Signatures. In *Symposium on Network and Distributed Systems Security (NDSS 2000)*, pages 143–154, 2000.

[293] H. Krawczyk and T. Rabin. Chameleon hashing and signatures. URL `http://www.google.com/patents/US6108783` [last accessed: Sept. 2017], 2000. US Patent – US 6108783 A.

[294] S. Krenn, K. Samelin, and D. Sommer. Stronger Security for Sanitizable Signatures. In *Proc. of International Workshop on Data Privacy Management (DPM 2015)*, volume 9481 of *LNCS*, pages 100–117. Springer, 2015.

[295] A. Kundu and E. Bertino. Structural Signatures for Tree Data Structures. In *Proc. of Int. Conference on Very Large Data Bases (VLDB 2008)*, pages 138–150. ACM, 2008. URL `http://www.vldb.org/pvldb/1/1453876.pdf` [last accessed: Sep. 2017].

[296] A. Kundu and E. Bertino. CERIAS Tech Report 2009-1 Leakage-Free Integrity Assurance for Tree Data Structures, 2009.

[297] A. Kundu and E. Bertino. How to authenticate graphs without leaking. In *Proc. of International Conference on Extending Database Technology (EDBT 2010)*, volume 426, pages 609–620. ACM, 2010. URL `http://doi.acm.org/10.1145/1739041.1739114` [last accessed: Sep. 2017].

[298] A. Kundu and E. Bertino. Structural signatures: How to authenticate trees without leaking. Technical report, Purdue University, 2010.

[299] A. Kundu and E. Bertino. Privacy-preserving authentication of trees and graphs. *International Journal of Information Security*, 12(6):467–494. Springer, 2013.

[300] A. Kundu, M. J. Atallah, and E. Bertino. Leakage-free redactable signatures. In *Proc. of ACM Conference on Data and Application Security and Privacy (CODASPY 2012)*, pages 307–316. ACM, 2012. URL `http://doi.acm.org/10.1145/2133601.2133639` [last accessed: Sep. 2017].

[301] A. Kung, F. Kargl, S. Suppan, J. Cuellar, H. C. Pöhls, A. Kapovits, N. Notario, and Y. S. Martin. A Privacy Engineering Framework for the Internet of Things. In *Proc. of International Conference on Computers, Privacy and Data Protection 2016 (CDPD 2016)*, volume 36 of *LGTS*. Springer, 2016. URL `https://doi.org/10.1007/978-3-319-50796-5` [last accessed: Sep. 2017].

[302] R. Küsters, T. Truderung, and A. Vogt. Accountability: definition and relationship to verifiability. In *Proc. of ACM conference on Computer and Communications Security (CCS 2010)*, pages 526–535. ACM, 2010. URL `https://doi.org/10.1145/1866307.1866366` [last accessed: Sep. 2017].

[303] C. Laborde. *Electronic Signatures in International Contracts*, volume 4982. Peter Lang, 2010.

[304] J. Lai, X. Ding, and Y. Wu. Accountable Trapdoor Sanitizable Signatures. In *Proc. of International Conference on Information Security Practice and Experience (ISPEC 2013)*, volume 7863 of *LNCS*, pages 117–131. Springer, 2013. URL `https://link.springer.com/chapter/10.1007/978-3-642-38033-4_9` [last accessed: Sep. 2017].

[305] R. W. F. Lai, T. Zhang, S. S. M. Chow, and D. Schröder. Efficient Sanitizable Signatures Without Random Oracles. In *Proc. of European Symposium on Research in Computer Security (ESORICS 2016)*, volume 9878 of *LNCS*, pages 363–380. Springer, 2016. URL `http://dx.doi.org/10.1007/978-3-319-45744-4_18` [last accessed: Sep. 2017].

[306] P. Landrock and T. Pedersen. WYSIWYS? — What you see is what you sign? *Information Security Technical Report*, 3(2):55–61. Elsevier, 1998. URL `http://dx.doi.org/10.1016/S0167-4048(98)80005-8` [last accessed: Sep. 2017].

[307] T. Länger, H. C. Pöhls, and S. Ghernaouti. Selected Cloud Security Patterns to Improve End User Security and Privacy in Public Clouds. In *Proc. of Annual Privacy Forum (APF 2016)*, volume 9857 of *LNCS*, pages 115–132. Springer, 2016. URL `http://henrich.poehls.com/papers/2016_LaengerPoehlsGhernaouti_SecurityPatternsForClouds_ENISA-APF.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1007/978-3-319-44760-5_8` [last accessed: Sep. 2017].

[308] D. C. Latham. Department of defense trusted computer system evaluation criteria, 1985.

[309] P. Lazaratos. The Grammatical Interpretation of European Law. URL `http://translation.hau.gr/telamon/files/Lazaratos.pdf` [last accessed: Jan. 2016], 2007.

[310] Legion of the Bouncy Castle Inc. The Legion of the Bouncy Castle. URL `https://bouncycastle.org/specifications.html` [last accessed: Oct. 2016], 2016.

[311] J. Leyden. Article in the register 'Last year's ICO fines would be 79 times higher under GDPR'. URL `http://www.theregister.co.uk/2017/04/28/ico_fines_post_gdpr_analysis/` [last accessed: Apr. 2017], 2017.

[312] J. Li, N. Li, and R. Xue. Universal Accumulators with Efficient Nonmembership Proofs. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2007)*, volume 4521 of *LNCS*, pages 253–269. Springer, 2007.

[313] X. Li, K. Chen, and S. Li. Designated-verifier proxy signatures for e-commerce from bilinear pairings. In *Proc. of the Int. Conf. on Computer Communication*, pages 1249–1252, 2004.

[314] S. Lim, E. Lee, and C.-M. Park. A short redactable signature scheme using pairing. *Security and Communication Networks*, 5(5):523–534. Wiley Online Library, 2011.

[315] A. Lioy, G. Ramunno, M. D. Aime, and M. Pala. Motivations for a theoretical approach to WYSIWYS. In *Proc. of IFIP International Conference on Communications and Multimedia Security (CMS 2005)*, volume 3677 of *LNCS*, pages 289–290. Springer, 2005.

[316] H. Lipmaa. Secure Accumulators from Euclidean Rings without Trusted Setup. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2012)*, volume 7341 of *LNCS*, pages 224–240. Springer, 2012.

[317] M. A. Lisovich, D. K. Mulligan, and S. B. Wicker. Inferring Personal Information from Demand-Response Systems. *IEEE Security and Privacy*, 8(1):11–20. IEEE Educational Activities Department, 2010. URL http://dx.doi.org/10.1109/MSP.2010.40 [last accessed: Sep. 2017].

[318] B. Liu, J. Lu, and J. Yip. XML data integrity based on concatenated hash function. *International Journal of Computer Science and Information Security*, 1(1). IJCSIS, 2009. URL https://sites.google.com/site/ijcsis/special-issue-may2009/18040921paper04.pdf [last accessed: Sep. 2017].

[319] D. Loebenberger and M. Nüsken. Notions for RSA integers. *International Journal of Applied Cryptography*, 3(2):116–138. 2014. URL http://dx.doi.org/10.1504/IJACT.2014.062723 [last accessed: Sep. 2017].

[320] J. Lopez, R. Oppliger, and G. Pernul. Why have public key infrastructures failed so far? *Journal of Internet Research*, 15(5):544–556. Emerald, 2005. URL http://dx.doi.org/10.1108/10662240510629475 [last accessed: Sep. 2017].

[321] T. Lorünser, C. B. Rodriguez, D. Demirel, S. Fischer-Hübner, T. Gross, T. Länger, M. des Noes, H. C. Pöhls, B. Rozenberg, and D. Slamanig. Towards a New Paradigm for Privacy and Security in Cloud Services. In *Proc. of Cyber Security and Privacy EU Forum (CSP Forum 2015)*, volume 530 of *CCIS*, pages 14–25. Springer, 2015. URL https://doi.org/10.1007/978-3-319-25360-2_2 [last accessed: Sep. 2017].

[322] T. Lorünser, D. Slamanig, T. Länger, and H. C. Pöhls. PRISMACLOUD Tools: A Cryptographic Toolbox for Increasing Security in Cloud Services. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2016)*, pages 733–741, 2016. URL http://henrich.poehls.com/papers/2016_LoruenserSlamanigLaengerPoehls_PRISMCLOUD-Architecture_SECPID_ARES.pdf [last accessed: Sep. 2017]. URL https://doi.org/10.1109/ARES.2016.62 [last accessed: Sep. 2017].

[323] R. Lumb, D. Treat, and O. Jelf. EDITING THE UNEDITABLE BLOCKCHAIN – Why distributed ledger technology must adapt to an imperfect world. URL https://newsroom.accenture.com/content/1101/files/Cross-FSBC.pdf [last accesssed: Oct. 2017], 2016.

[324] O. Lynskey, N. Robinson, and M. Greenberg. e-Discovery and legal frameworks governing Privacy and Data Protection in European countries. URL http://www.ftitechnology.com/resources/white-papers/rand-global-privacy-and-data-protection-part-1, 2010.

[325] J. Ma, J. Liu, M. Wang, and W. Wu. An Efficient and Secure Design of Redactable Signature Scheme with Redaction Condition Control. In *Proc. of International Conference on Green, Pervasive, and Cloud Computing (GPC 2017)*, volume 10232 of *LNCS*, pages 38–52. Springer, 2017. URL http://dx.doi.org/10.1007/978-3-319-57186-7_4 [last accessed: Sep. 2017].

[326] C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, and R. Metz. Reference Model for Service Oriented Architecture 1.0. *OASIS Standard*. 2006.

[327] M. Mambo and E. Usuda, K.and Okamoto. Proxy signatures for delegating signing operation. In *Proc. of ACM conference on Computer and Communications Security (CCS 1996)*, pages 48–57. ACM, 1996. URL http://doi.acm.org/10.1145/238168.238185 [last accessed: Sep. 2017].

[328] M. Mambo, K. Usuda, and E. Okamoto. Proxy signatures: Delegation of the power to sign messages. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 79(9):1338–1354. The Institute of Electronics, Information and Communication Engineers, 1996.

[329] J. Marly and H. Habermann. BGH Beschl. v. 14.5.2013 – VI ZB 7/13. *NJW Neue juristische Wochenzeitung*, page 2034. C.H.BECK, 2013.

[330] S. Mason. *Electronic Signatures in Law: Fourth Edition*. School of Advanced Study, University of London, 4th edition, 2017. URL http://dx.doi.org/10.14296/117.9781911507017 [last accessed: Sep. 2017].

[331] U. M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66. Springer, 1992. ISSN 1432-1378. URL http://dx.doi.org/10.1007/BF00191321 [last accessed: Sep. 2017].

[332] E. McCallister, T. Grance, and K. Scarfone. NIST Special Publication 800-122: Guide to Protecting the Confidentiality of Personally Identifiable Information (PII). URL http://doi.org/10.6028/NIST.SP.800-122 [last accessed: Jan. 2016], 2010.

[333] M. McIntosh and P. Austel. XML signature element wrapping attacks and countermeasures. In *Proc. of Workshop on Secure Web Services (SWS 2005)*, pages 20–27. ACM, 2005. doi: http://doi.acm.org/10.1145/1103022.1103026.

[334] S. Meissner, D. Dobre, M. Thoma, and G. Martin. Internet of Things Architecture IoT-A Project Deliverable D2. 1–Resource Description Specification. URL http://www.meet-iot.eu/deliverables-IOTA/D2_1.pdf [last accessed: Jan 2017], 2012.

[335] Meister. Einwilligung bei juristischen Personen im Untreuekontext. *Bucerius Law Journal*, (2):37–45. Bucerius Law School, 2012.

[336] Member States. Consolidated Version of the Treat on European Union. *Official Journal* , OJ C 83 of 30.3.2010:13–45. 2010.

[337] A. J. Menezes, P. C. V. Oorschot, S. A. Vanstone, and R. L. Rivest. *Handbook of Applied Cryptography*. CRC Press, 1st edition, 1997. ISBN 978-0849385230.

[338] R. Merkle. A Certified Digital Signature. In *Proc. of Advances in Cryptology – CRYPTO' 89*, volume 435 of *LNCS*, pages 218–238. Springer, 1989. URL https://doi.org/10.1007/0-387-34805-0_21 [last accessed: Sep. 2017].

[339] S. Micali and R. L. Rivest. Transitive Signature Schemes. In *Proc. of Topics in Cryptology – CT-RSA 2002*, volume 2271 of *LNCS*, pages 236–243. Springer, 2002.

[340] E. Michiels. ISO/IEC 10181-6: Information technology — Open Systems Interconnection — Security frameworks for open systems: Integrity framework. *ISO Geneve, Switzerland*. 1996.

[341] Y. Ming, X. Shen, and Y. Peng. Identity-based sanitizable signature scheme in the standard model. In *Proc. of International Conference on Information Computing and Applications (ICICA 2010)*, volume 106 of *CCIS*, pages 9–16. Springer, 2010. URL http://dx.doi.org/10.1007/978-3-642-16336-4_2 [last accessed: Sep. 2017].

[342] Y. Ming, X. Shen, and Y. Peng. Provably security identity-based sanitizable signature scheme without random oracles. *Journal of Software*, 6(10):1890–1897. Internation Academy Publishing, 2011.

[343] K. Miyazaki. Redactable digital signatures for secure and easy-to-use digital document systems. URL http://wayback.archive.org/web/20110708080212/http://www.hitachi.com/rd/yrl/people/suminuri/index.html [last accessed: Jan. 2016], 2008.

[344] K. Miyazaki and G. Hanaoka. Invisibly sanitizable digital signature scheme. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 91(1):392–402. The Institute of Electronics, Information and Communication Engineers, 2008.

[345] K. Miyazaki and Y. Hatano. Electronic document authenticity guarantee method, and electronic document disclosure system. URL http://www.google.la/patents/US7941667 [last accessed: Jan. 2017], 2011. US Patent – US 7941667 B2.

[346] K. Miyazaki, S. Susaki, M. Iwamura, T. Matsumoto, R. Sasaki, and H. Yoshiura. Digital documents sanitizing problem. Technical Report ISEC 2003-20, The Institute of Electronics, Information and Communication Engineers (IEICE), 2003. (in Japanese language).

[347] K. Miyazaki, M. Iwamura, T. Matsumoto, R. Sasaki, H. Yoshiura, S. Tezuka, and H. Imai. Digitally Signed Document Sanitizing Scheme with Disclosure Condition Control. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 88-A(1):239–246. IEICE, 2005.

[348] K. Miyazaki, G. Hanaoka, and H. Imai. Digitally signed document sanitizing scheme based on bilinear maps. In *Proc. of ACM Symposium on Information, Computer and Communications Security (ASIA CCS 2006)*, pages 343–354. ACM, 2006. URL http://doi.acm.org/10.1145/1128817.1128868 [last accessed: Sep. 2017].

[349] K. Miyazaki, G. Hanaoka, and H. Imai. Invisibly sanitizable digital signature scheme. *IEICE Transactions*, 91-A(1):392–402. IEICE, 2008. URL http://dx.doi.org/10.1093/ietfec/e91-a.1.392 [last accessed: Sep. 2017].

[350] G. Moldovan, E. Z. Tragos, A. Fragkiadakis, H. C. Pöhls, and D. Calvo. An IoT Middleware for Enhanced Security and Privacy: The RERUM Approach. In *Proc. of IFIP International Conference on New Technologies, Mobility and Security (NTMS 2016)*, pages 1–5. IEEE, 2016. URL http://henrich.poehls.com/papers/2016_Moldovan-et-al_IoT-middleware-for-enhanced-sec-and-priv-RERUM_NTMS.pdf [last accessed: Sep. 2017]. URL https://doi.org/10.1109/NTMS.2016.7792434 [last accessed: Sep. 2017].

[351] A. Molina-Markham, P. Shenoy, K. Fu, E. Cecchet, and D. Irwin. Private memoirs of a smart meter. In *Proc. of 2nd ACM BuildSys '10*, pages 61–66. ACM, 2010. ISBN 978-1-4503-0458-0. URL http://doi.acm.org/10.1145/1878431.1878446 [last accessed: Sep. 2017].

[352] G. Monakova, C. Severin, A. D. Brucker, U. Flegel, and A. Schaad. Monitoring Security and Safety of Assets in Supply Chains. In *Proc. of 7th Security Research Conference, Future Security 2012*, volume 318 of *CCIS*, pages 9–20. Springer, 2012. URL https://link.springer.com/chapter/10.1007/978-3-642-33161-9_3 [last accessed: Sep. 2017].

[353] M. Mössinger, B. Petschkuhn, J. Bauer, R. C. Staudemeyer, M. Wojcik, and H. C. Pöhls. Towards quantifying the cost of a secure IoT: Overhead and energy consumption of ECC signatures on an ARM-based device. In *Proc. of workshop on IoT-SoS: Internet of Things Smart Objects and Services (WOWMOM SOS-IOT 2016)*, pages 1–6. IEEE, 2016. URL http://henrich.poehls.com/papers/2016_Moessinger_et_al-Towards_quantifying_the_cost_of_a_secure_IoT.pdf [last accessed: Sep. 2017]. URL https://doi.org/10.1109/WoWMoM.2016.7523559 [last accessed: Sep. 2017].

[354] National Institute of Standards and Technology (NIST). PUB FIPS 199. Standards for Security Categorization of Federal Information and Information Systems, 2004.

[355] National Institute of Standards and Technology (NIST). PUB FIPS 186-4. Digital signature standard (DSS), 2013.

[356] National Institute of Standards and Technology (NIST). NIST: FISMA Overview. URL http://csrc.nist.gov/groups/SMA/fisma/overview.html, 2014.

[357] R. M. Needham. 'Address' in 'The Marshall symposium — the information revolution in midstream: an Anglo-American perspective'. Rackham Schhol of Graduate Studies, University of Michigan. URL https://www.m-chair.net/images/documents/lectures/2015SS/MOB2/Marshall_Symposium_Address_Roger_Needham.pdf [last accessed: Jan. 2018], 1998.

[358] R. Nojima, J. Tamura, Y. Kadobayashi, and H. Kikuchi. A storage efficient redactable signature in the standard model. In *Proc. of International Conference on Information Security (ISC 2009)*, pages 326–337. Springer, 2009. URL http://dx.doi.org/10.1007/978-3-642-04474-8_26 [last accessed: Sep. 2017].

[359] K. Nyberg. Fast accumulated hashing. In *Proc. of Fast Software Encryption (FSE '96)*, volume 1039 of *LNCS*, pages 83–87. Springer, 1996.

[360] OECD. The OECD Privacy Framework. URL http://oecd.org/sti/ieconomy/oecd_privacy_framework.pdf [last accessed: Jan. 2017], 2013.

[361] OLG Brandenburg. Urteil vom 15.06.2006 – Az. 3U 179/04. URL http://www.olg.brandenburg.de/sixcms/media.php/4250/3U179-04.pdf [last accessed: Jan. 2018], 2006.

[362] OLG Hamburg. Urteil vom 15.04.2010 – Az. 5 U 106/08. URL https://openjur.de/u/51702.html [last accessed: Jan. 2016], 2010.

[363] OLG München. Urteil vom 04.06.2012 – Az. 19 U 771/12. URL http://openjur.de/u/498795.html [last accessed: Feb. 2016], 2012.

[364] J. Ølnes and A. B. Seip. On long-term storage of digitally signed documents. In *Towards the Knowledge Society*, pages 115–130. Springer, 2003.

[365] C. Orthacker, M. Centner, and C. Kittl. Qualified Mobile Server Signature. In *Proc. of IFIP TC-11 International Information Security Conference (IFIP SEC 2010)*, pages 103–111. Springer, 2010. URL http://dx.doi.org/10.1007/978-3-642-15257-3_10 [last accessed: Sep. 2017].

[366] L. Owens, A. Duffy, and T. Dowling. An Identity Based Encryption system. In *Proc. of international symposium on Principles and practice of programming in Java (PPPJ '04)*, volume 91 of *International Conference Proceeding Series*, pages 154–159. ACM, 2004. URL https://dl.acm.org/citation.cfm?id=1071565.1071594 [last accessed: Sep. 2017].

[367] Parliament of the United Kingdom. Freedom of Information Act 2000. URL http://www.legislation.gov.uk/ukpga/2000/36/pdfs/ukpga_20000036_en.pdf, 2000.

[368] Parliament of the United Kingdom. Electronic Communications Act 2000. URL http://www.legislation.gov.uk/ukpga/2000/7/pdfs/ukpga_20000007_en.pdf, 2000.

[369] J. Pejaś and M. Zawalich. Visual Cryptography Methods as a Source of Trustworthiness for the Signature Creation and Verification Systems. In *Proc. of Advances in Information Processing and Protection*, pages 225–239. Springer, 2007. URL https://doi.org/10.1007/978-0-387-73137-7_20 [last accessed: Sep. 2017].

[370] S. Peters. Sanitizable Signatures on Smart Card – An Implementation on Java Card, Performance Evaluation and Analysis of Suitability for Selected Sanitizable Signature Schemes. Master thesis, IT-Security Group, University of Passau, 2012.

[371] A. Pfitzmann and M. Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management, 2009. URL http://dud.inf.tu-dresden.de/Anon_Terminology.shtml [last accessed: Sep. 2017]. v0.32.

[372] B. Pfitzmann. Sorting Out Signature Schemes. Technical Report ISSN 0941-3014, University of Hildesheim, 1993.

[373] B. Pfitzmann and M. Waidner. Composition and integrity preservation of secure reactive systems. In *Proc. of ACM conference on Computer and communications security (CCS 2000)*, pages 245–254. ACM, 2000.

[374] H. C. Pöhls. Authenticity and Revocation of Web Content using Signed Microformats and PKI. Technical Report B-276-07, University of Hamburg, Department of Informatics, Hamburg, Germany, 2007. URL http://edoc.sub.uni-hamburg.de/informatik/volltexte/2009/16/pdf/B_276_07.pdf [last accessed: Sep. 2017].

[375] H. C. Pöhls. Verifiable and revocable expression of consent to processing of aggregated personal data. In *Proc. of International Conference on Information and Communications Security (ICICS 2008)*, volume 5308 of *LNCS*, pages 279–293. Springer, 2008. URL http://dx.doi.org/10.1007/978-3-540-88625-9_19 [last accessed: Sep. 2017]. URL http://henrich.poehls.com/papers/2008_Poehls_RevocableExpressionOfConsent_ICICS2008.pdf [last accessed: Sep. 2017].

[376] H. C. Pöhls. ConCert: Content revocation using certificates. In *Proc. of the Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik (GI Sicherheit 2008)*, volume 128 of *GI-Edition Lecture Notes in Informatics (LNI)*, pages 149–162. GI, 2008. URL http://henrich.poehls.com/papers/2008_poehls_GI-Sicherheit_Concert-Content-revocation-using-certificates.pdf [last accessed: Sep. 2017]. URL http://137.226.34.227/dblp/db/conf/sicherheit/sicherheit2008.html [last accessed: Sep. 2017].

[377] H. C. Pöhls. Why showing one TLS certificate is not enough? towards a browser feedback for multiple TLS certificate verifications. In *Proc. of the Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik (GI Sicherheit 2010)*, volume 170 of *GI-Edition Lecture Notes in Informatics (LNI)*, pages 265–276. GI, 2010. URL http://henrich.poehls.com/papers/2010_Poehls_Show_Multiple_SSL_Certificate_Verifications_GI-Sicherheit.pdf [last accessed: Sep. 2017]. https://subs.emis.de/LNI/Proceedings/Proceedings170/265.pdf.

[378] H. C. Pöhls. No Integrity, thanks — Verifiable Explicit Consented Undecidability of Integrity. Short paper presentation at the Information Security Conference 2012 (ISC'12), 2012. URL http://web.sec.uni-passau.de/members/henrich/talks/Poehls_2012_ISC_No-Integrity-thanks.pdf [last accessed: Sep. 2017].

[379] H. C. Pöhls. Contingency Revisited: Secure Construction and Legal Implications of Verifiably Weak Integrity. In *Proc. of IFIP WG 11.11 International Conference on Trust Management (IFIPTM 2013)*, volume 401 of *IFIPAICT*, pages 136 – 150. Springer, 2013. URL http://dx.doi.org/10.1007/978-3-642-38323-6_10 [last accessed: Sep. 2017].

[380] H. C. Pöhls. JSON Sensor Signatures (JSS): End-to-End Integrity Protection from Constrained Device to IoT Application. In *Proc. of the Workshop on Extending Seamlessly to the Internet of Things (esIoT) at the IMIS-2012 International Conference (IMIS 2015)*, pages 306 – 312. IEEE, 2015. URL http://henrich.poehls.com/papers/2015_Poehls-JSONSensorSignatures_esIoT.pdf [last accessed: Sep. 2017]. URL http://dx.doi.org/10.1109/IMIS.2015.48 [last accessed: Sep. 2017].

[381] H. C. Pöhls and F. Höhne. The Role of Data Integrity in EU Digital Signature Legislation - Achieving Statutory Trust for Sanitizable Signature Schemes. In *Revised Selected Papers from the 7th International Workshop on Security and Trust Management (STM 2011)*, volume 7170 of *LNCS*, pages 175–192. Springer, 2011. URL http://dx.doi.org/10.1007/978-3-642-29963-6_13 [last accessed: Sep. 2017].

[382] H. C. Pöhls and F. Höhne. Sticky Signatures: Legal Advantages of Redactable Signatures and Credentials in the Food Supply Chain. In *Proc. of Interdisciplinary Conference on Current Issues in IT Security 2012*, Interdisciplinary Series of the Max Planck Institute for Foreign and International Criminal Law (MPICC). Dunker & Humblot, Berlin, 2012.

[383] H. C. Pöhls and M. Karwe. Redactable signatures to control the maximum noise for differential privacy in the smart grid. In *Proc. of International Workshop on Smart Grid Security (SmartGridSec 2014)*, volume 8448 of *LNCS*, pages 79–93. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-319-10329-7_6` [last accessed: Sep. 2017].

[384] H. C. Pöhls and T. Länger. Einsetzbare Kryptografie für die Cloud. *Zeitschrift für Datenrecht und Informationssicherheit (digma)*, 17(1):78–81. Schulthess Juristische Medien, 2017.

[385] H. C. Pöhls and B. Petschkuhn. Towards compactly encoded signed IoT messages. In *Proc. of IEEE International Workshop on Computer-Aided Modeling Analysis and Design of Communication Links and Networks (IEEE CAMAD 2017)*, pages 1–6. IEEE, 2017. URL `https://doi.org/10.1109/CAMAD.2017.8031622` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2017_PoehlsPetschkuhn_IoT_signature_encoding_CAMAD.pdf` [last accessed: Sep. 2017].

[386] H. C. Pöhls and J. Posegga. Smartcard firewalls revisited. In *Proc. of IFIP WG 8.8/11.2 International Conference on Smart Card Research and Advanced Applications (CARDIS 2006)*, volume 3928 of *LNCS*, pages 179 – 191. Springer, 2006. URL `http://www.springerlink.com/content/t42644k876vg612k/` [last accessed: Sep. 2017].

[387] H. C. Pöhls and K. Samelin. On updatable redactable signatures. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2014)*, volume 8479 of *LNCS*, pages 457–475. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-319-07536-5_27` [last accessed: Sep. 2017].

[388] H. C. Pöhls and K. Samelin. Accountable Redactable Signatures. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2015)*, pages 60 – 69. IEEE, 2015. URL `http://dx.doi.org/10.1109/ARES.2015.10` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2015_Poehls-Samelin_Accountable-Redactable-Signatures_ARES15.pdf` [last accessed: Sep. 2017].

[389] H. C. Pöhls and L. Westphal. Die "Untiefen" der neuen XML-basierten Dokumentenformate. In *Proc. of DFN CERT Workshop Sicherheit in vernetzten Systemen*, 2008. URL `https://www.dfn-cert.de/dokumente/workshop/2008/poehls.pdf` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2008_Poehls_Westphal_2008_DFN-CERT-WS_Untiefen_der_XML-Dokumentenformate.pdf` [last accessed: Sep. 2017].

[390] H. C. Pöhls, A. Bilzhause, K. Samelin, and J. Posegga. Sanitizable Signed Privacy Preferences for Social Networks. In *Proc. of the GI Workshop on Privacy and Identity Management for Communities - Communities for Privacy and Identity Management (DICCDI 2011)* , volume 192 of *GI-Edition Lecture Notes in Informatics (LNI)*. GI, 2011. URL `http://www.user.tu-berlin.de/komm/CD/paper/090212.pdf` [last accessed: Sep. 2017].

[391] H. C. Pöhls, K. Samelin, and J. Posegga. Sanitizable Signatures in XML Signature - Performance, Mixing Properties, and Revisiting the Property of Transparency. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2011)*, volume 6715 of *LNCS*, pages 166–182. Springer, 2011. URL `http://dx.doi.org/10.1007/978-3-642-21554-4_10` [last accessed: Sep. 2017].

[392] H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Length-Hiding Redactable Signatures from One-Way Accumulators in $\mathcal{O}(n)$ (MIP-1201). Technical Report MIP-1201, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP-1201.pdf` [last accessed: Sep. 2017].

[393] H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Transparent Mergeable Redactable Signatures with Signer Commitment and Applications (MIP-1206). Technical Report MIP-1206, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/TR_MIP1206.pdf` [last accessed: Sep. 2017].

[394] H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Flexible Redactable Signature Schemes for Trees — Extended Security Model and Construction. In *Proc. of the International Conference on Security and Cryptography (SECRYPT 2012)*, pages 113–125. SciTePress, 2012. URL `http://t1p.de/SECRYPT2012` [last accessed: Sep. 2017].

[395] H. C. Pöhls, S. Peters, K. Samelin, J. Posegga, and H. de Meer. Malleable Signatures for Resource Constrained Platforms. In *Proc. of IFIP WG 11.2 International Workshop in Information Security Theory and Practice (WISTP 2013)*, number 7886 in LNCS, pages 18–33. Springer, 2013. URL `https://dx.doi.org/10.1007/978-3-642-38530-8_2` [last accessed: Sep. 2017].

[396] H. C. Pöhls, V. Angelakis, S. Suppan, K. Fischer, G. Oikonomou, E. Z. Tragos, R. D. Rodriguez, and T. Mouroutis. RERUM: Building a Reliable IoT upon Privacy- and Security- enabled Smart Objects. In *Proc. of the Workshop on Internet of Things Communications and Technologies (IEEE WCNC 2014)*, pages 122–127. IEEE, 2014. URL `http://dx.doi.org/10.1109/WCNCW.2014.6934872` [last accessed: Sep. 2017].

[397] H. C. Pöhls, B. Petschkuhn, J. Rückert, and M. Mössinger. Aggregation and perturbation in practice: Case-study of privacy, accuracy and performance. In *IEEE International Workshop on Computer-Aided Modeling Analysis and Design of Communication Links and Networks (IEEE CAMAD 2014)*, pages 183–187. IEEE, 2014. URL `http://dx.doi.org/10.1109/CAMAD.2014.7033231` [last accessed: Sep. 2017].

[398] PONS GmbH, Stuttgart. PONS Online Dictionary. URL `http://en.pons.eu` [last accessed: Sep. 2016], 2016.

[399] U. Pordesch. *Die elektronische Form und das Präsentationsproblem*. Nomos, 2003. ISBN 978-3-8329-0069-4.

[400] J. Posegga, D. Heckmann, H. C. Pöhls, and F. Höhne. Abschlussbericht Teilvorhaben FKZ 13N10966 des Projektes ReSCUeIT: Trusted Lawful Supply Chains (TruLy), 2014. URL `http://edok01.tib.uni-hannover.de/edoks/e01fb14/804985758.pdf` [last accessed: Sep. 2017].

[401] W. V. O. Quine. *Word and Object*. MIT Press, 1964. ISBN 9780262670012.

[402] K. Radke, C. Boyd, J. Gonzalez Nieto, and M. Brereton. Ceremony analysis: Strengths and weaknesses. In *Proc. of IFIP TC 11 International Information Security Conference (SEC 2011)*, volume 354 of *IFIPAICT*, pages 104–115. Springer, 2011. URL `https://doi.org/10.1007/978-3-642-21424-0_9` [last accessed: Sep. 2017].

[403] W. Rankl and W. Effing. *Handbuch der Chipkarten (3rd edition)*. Carl Hanser Verlag, 3rd edition, 1999. ISBN 978-3-446-21115-5.

[404] A. Rayes and S. Salam. *Internet of Things–From Hype to Reality*. Springer, 2017. ISBN 978-3-319-44858-9.

[405] P. Rechenberg and G. Pomberger. *Informatik Handbuch*. Hanser Verlag, 4. edition, 2006. ISBN 978-3-446-40185-3.

[406] E. Reilly. *Concise Encyclopedia of Computer Science*. Wiley, 2004. ISBN 9780470090954. URL `http://books.google.de/books?id=5Jaa1BVverIC` [last accessed: Sep. 2017].

[407] R. L. Rivest. Two signature schemes. Talk given October 17, 2000 at Cambridge University. `http://people.csail.mit.edu/rivest/pubs.html` [last accessed: Jan. 2017], 2000.

[408] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 26(1):96–99. ACM, 1983. URL `http://doi.acm.org/10.1145/357980.358017` [last accessed: Sep. 2017].

[409] A. Roßnagel. Die fortgeschrittene elektronische Signatur. *MMR - MultiMedia und Recht*, pages 164–170. C.H.BECK, 2003.

[410] A. Roßnagel. Fremderzeugung von qualifizierten Signaturen? - Ein neues Geschäftsmodell und seine Rechtsfolgen. *MMR - MultiMedia und Recht*, page 22. C.H.BECK, 2008.

[411] A. Roßnagel. Das De-Mail-Gesetz – Grundlage für mehr Rechtssicherheit im Internet. *NJW Neue juristische Wochenzeitung*, pages 1473–1478. C.H.BECK, 2011.

[412] A. Roßnagel. Neue Regeln für sichere elektronische Transaktionen. In *NJW Neue juristische Wochenzeitung*, volume 51, page 3686. C.H.BECK, 2014.

[413] A. Roßnagel. Der Anwendungsvorrang der eIDAS-Verordnung – Welche Regelungen des deutschen Rechts sind weiterhin für elektronische Signaturen anwendbar? *MMR - MultiMedia und Recht*, page 359. C.H.BECK, 2015.

[414] A. Roßnagel. *Das Recht der Vertrauensdienste: Die eIDAS-Verordnung in der deutschen Rechtsordnung*. Nomos, 2016. ISBN 978-3-8452-7877-3.

[415] A. Roßnagel and S. Fischer-Dieskau. Elektronische Dokumente als Beweismittel - Neufassung der Beweisregelungen durch das Justizkommunikationsgesetz. *NJW Neue juristische Wochenzeitung*, 80(12):806. C.H.BECK, 2006.

[416] A. Roßnagel and M. Nebel. Policy Paper: Die neue Datenschutzgrundverordnung. Ist das Datenschutzrecht nun für heutige Herausforderungen gerüstet? In *Forum Privatheit und selbstbestimmtes Leben in der digitalen Welt*. Forum Privatheit und selbstbestimmtes Leben in der digitalen Welt. Karlsruhe., 2016.

[417] A. Roßnagel, G. Hornung, M. Knopp, and D. Wilke. De-Mail und Bürgerportale. *Datenschutz und Datensicherheit - DuD*, 33(12):728. Springer, 2009. URL `https://doi.org/10.1007/s11623-009-0193-1` [last accessed: Sep. 2017].

[418] H. Rossnagel. Mobile qualified electronic signatures and certification on demand. In *EuroPKI 2004*, pages 274–286. Springer, 2004.

[419] M. Rost and K. Bock. Privacy By Design und die Neuen Schutzziele. *Datenschutz und Datensicherheit - DuD*, 35(1):30–35. Springer, 2011. English Version can be found in [420].

[420] M. Rost and K. Bock. Privacy By Design and the New Protection Goals. URL `http://www.datenschutzgeschichte.de/pub/privacy/BockRost_PbD_DPG_en_v1f.pdf` [last accessed: Jan. 2018], 2011.

[421] M. Rost and A. Pfitzmann. Datenschutz-Schutzziele — revisited. *Datenschutz und Datensicherheit - DuD*, 33(6):353–358. Springer, 2009. URL `http://dx.doi.org/10.1007/s11623-009-0072-9` [last accessed: Sep. 2017].

[422] RSA Laboratories. PKCS #1 v2.2: RSA Cryptography Standard. URL `http://https://www.emc.com/collateral/white-papers/h11300-pkcs-1v2-2-rsa-cryptography-standard-wp.pdf` [last accessed: Jul. 2017], 2012.

[423] D. Ruiz and E. al. RERUM Deliverable D3.1 - Enhancing the autonomous smart objects and the overall system security of IoT based Smart Cities, 2015. URL `https://bscw.ict-rerum.eu/pub/bscw.cgi/d14540/RERUMdeliverableD3_1.pdf` [last accessed: Sep. 2017].

[424] H. Rüßmann. Das Beweisrecht elektronischer Dokumente. In *JurPC Web-Dok*, page 3212. Maximilian Herberger, 1995.

[425] K. Samelin, H. C. Pöhls, A. Bilzhause, J. Posegga, and H. de Meer. On Structural Signatures for Tree Data Structures. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2012)*, volume 7341 of *LNCS*, pages 171–187. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-31284-7_11` [last accessed: Sep. 2017].

[426] K. Samelin, H. C. Pöhls, A. Bilzhause, J. Posegga, and H. de Meer. Redactable Signatures for Independent Removal of Structure and Content. In *Proc. of International Conference on Information Security Practice and Experience (ISPEC 2012)*, volume 7232 of *LNCS*, pages 17–33. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-29101-2_2` [last accessed: Sep. 2017].

[427] K. Samelin, H. C. Pöhls, J. Posegga, and H. de Meer. Redactable vs. Sanitizable Signatures (MIP-1208). Technical Report MIP-1208, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP-1208_neu.pdf` [last accessed: Sep. 2017].

[428] K. Samelin, H. C. Pöhls, J. Posegga, and H. de Meer. Block-level Accountability for Transparent Sanitizable Signatures (MIP-1209). Technical report, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP1209.pdf` [last accessed: Sep. 2017].

[429] T. Sander. Efficient accumulators without trapdoor extended abstracts. In *Proc. of International Conference on Information and Communication Security (ICICS '99)*, volume 1726 of *LNCS*, pages 252–262. Springer, 1999.

[430] R. S. Sandhu. Lattice-Based Access Control Models. *Computer*, 26:9–19. IEEE, 1993. URL `http://portal.acm.org/citation.cfm?id=618985.619980` [last accessed: Sep. 2017].

[431] Schiphol Group. Schiphol Pass application form. URL `http://www.schiphol.nl/web/file?uuid=1f52accf-920f-4ec9-833b-68f71e26e30e` [last accessed: Oct. 2014], 2009.

[432] A. U. Schmidt. Signiertes XML und das Präsentationsproblem. *Datenschutz und Datensicherheit - DuD*, 24(3):153–158. Springer, 2000. URL `www.ethemba.novalyst.de/docs/DuD24_153_schmidt.pdf` [last accessed: Sep. 2017].

[433] B. Schneier. *Secret and Lies*. John Wiley and Sons, 2000. ISBN 978-0471253112.

[434] A. Schönke and H. Schröder. *Strafgesetzbuch, Kommentar*. C.H.BECK, 28 edition, 2010. ISBN 978-3-406-60404-1.

[435] K.-H. Schramm. *Münchener Kommentar zum BGB*, volume 6. C.H.BECK, 2012. ISBN 978-3-406-61466-8.

[436] SCM PC-Card, GmbH. SCR335/SCR335 v2.0 Datasheet, 2011. URL `http://www.scm-pc-card.de/index.php?page=product&function=product_datasheet&lang=en&product_id=826` [last accessed: Sep. 2017].

[437] Senate and House of Representatives of the United States of America in Congress assembled. American Recovery and Reinvestment Act of 2009. URL `http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=111_cong_bills&docid=f:h1enr.pdf` [last accessed: Jan. 2017], 2009.

[438] R. Shirey. Internet Security Glossary. RFC 4949, RFC Editor, 2007. URL `http://www.rfc-editor.org/rfc/rfc4949.txt` [last accessed: Sep. 2017].

[439] P. W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5):1484–1509. SIAM, 1997. URL `http://dx.doi.org/10.1137/S0097539795293172` [last accessed: Sep. 2017].

[440] S. Simitis, editor. *Kommentar zum BDSG*. Nomos, 8 edition, 2014. ISBN 978-3-8487-0593-1.

[441] D. Slamanig and S. Rass. Generalizations and extensions of redactable signatures with applications to electronic healthcare. In *Proc. of IFIP International Conference on Communications and Multimedia Security (CMS 2010)*, volume 6109 of *LNCS*, pages 201–213. Springer, 2010. URL `http://dx.doi.org/10.1007/978-3-642-13241-4_19` [last accessed: Sep. 2017].

[442] D. Slamanig and S. Rass. Generalizations and extensions of redactable signatures with applications to electronic healthcare. In *Communications and Multimedia Security*, pages 201–213. Springer, 2010.

[443] D. Slamanig and S. Rass. Redigierbare Digitale Signaturen. *Datenschutz und Datensicherheit - DuD*, 35(11): 757–762. Springer, 2011. ISSN 1862-2607. URL `http://dx.doi.org/10.1007/s11623-011-0181-0` [last accessed: Sep. 2017].

[444] D. Slamanig and C. Stingl. Disclosing verifiable partial information of signed CDA documents using generalized redactable signatures. In *Proc. of International Conference on e-Health Networking, Applications and Services (Healthcom 2009)*, pages 146–152. IEEE, 2009. URL `https://doi.org/10.1109/HEALTH.2009.5406190` [last accessed: Sep. 2017].

[445] D. Slamanig, K. Stranacher, and B. Zwattendorfer. User-centric identity as a service-architecture for eIDs with selective attribute disclosure. In *Proc. of ACM symposium on Access control models and technologies (SACMAT 2014)*, pages 153–164. ACM, 2014.

[446] A. Smith and J. B. MacKinnon. *The 100-mile diet: A year of local eating*. Vintage Canada, 2009.

[447] D. J. Solove. A Taxonomy of Privacy. *University of Pennsylvania Law Review*, 154:477. 2006.

[448] D. J. Solove. I've got nothing to hide and other misunderstandings of privacy. *San Diego Law Review*, 44: 745–772. HeinOnline, 2007.

[449] J. Somorovsky, A. Mayer, J. Schwenk, M. Kampmann, and M. Jensen. On Breaking SAML: Be Whoever You Want to Be. In *Proc. of the 21st USENIX Security Symposium*, pages 397–412. USENIX, 2012. ISBN 978-931971-95-9.

[450] H. Søndergaard and P. Sestoft. Referential transparency, definiteness and unfoldability. *Acta Informatica*, 27 (6):505–517. 1990. URL `http://dx.doi.org/10.1007/BF00277387` [last accessed: Sep. 2017].

[451] C. Sorge. The legal classification of identity-based signatures. *Journal: Computer Law & Security Review*, 30(2):126–136. 2014. URL `http://dx.doi.org/10.1016/j.clsr.2014.01.002` [last accessed: Sep. 2017].

[452] W. Stallings. *Network Security Essentials: Applications and Standards (3rd Edition)*. Prentice Hall, 2006. ISBN 0132380331.

[453] M. Stamp. *Information security: principles and practice*. John Wiley & Sons, 2011. ISBN 978-0470626399.

[454] State of California – Department of Water Rseources. Application for approval of plans and specifications for the construction or enlargement of a dam and reservoir. URL `http://www.water.ca.gov/publications/forms/3.pdf` [last accessed: Oct. 2017], 2009.

[455] R. Steinfeld, L. Bull, and Y. Zheng. Content extraction signatures. In *Proc. of International Conference on Information Security and Cryptology (ICISC 2001)*, volume 2288, pages 163–205. Springer, 2002. URL `https://doi.org/10.1007/3-540-45861-1_22` [last accessed: Sep. 2017].

[456] R. Steinfeld, L. Bull, H. Wang, and J. Pieprzyk. Universal designated-verifier signatures. In *Proc. of Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *LNCS*, pages 523–542. Springer, 2003. URL `https://doi.org/10.1007/978-3-540-40061-5_33` [last accessed: Sep. 2017].

[457] R. Steinfeld, L. Bull, and Y. Zheng. Content Extraction Signatures (updated). URL https://pdfs.semanticscholar.org/173d/9ea9064fe6f47238a143350299ee84bfa14d.pdf [last accessed: Sept. 2017], 2003.

[458] C. Strachey. Fundamental concepts in programming languages. *Higher-Order and Symbolic Computation*, 13(1):11–49. Springer, 2000. URL http://dx.doi.org/10.1023/A:1010000313106 [last accessed: Sep. 2017].

[459] K. Stranacher, V. Krnjic, and T. Zefferer. Trust and reliability for public sector data. In *Proc. of International Conference on e-Business and e-Government*, volume 73, pages 124 – 132, 2013.

[460] K. Stranacher, V. Krnjic, B. Zwattendorfer, and T. Zefferer. Evaluation and Assessment of Editable Signatures for Trusted and Reliable Public Sector Data. In *European Conference on e-Government (ECeG'13)*, pages 508–516. ACPI, 2013.

[461] M. Z. Stuart Haber, William Horne. Efficient Transparent Redactable Signatures with a Single Signature Invocation. Technical Report HPL-2014-R1, Hewlett-Packard Development Company, 2015.

[462] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570. World Scientific, 2002.

[463] K. W. Tan and R. H. Deng. Applying sanitizable signature to web-service-enabled business processes: Going beyond integrity protection. In *IEEE International Conference on Web Services (ICWS)*, pages 67–74. IEEE, 2009.

[464] Techtarget. WhatIs.com reference by Techtarget. URL http://whatis.techtarget.com/definition/transparent [last accessed: Feb. 2017], 2010.

[465] H. Tews and B. Jacobs. Performance issues of selective disclosure and blinded issuing protocols on java card. *Information Security Theory and Practice. Smart Devices, Pervasive Systems, and Ubiquitous Networks*, pages 95–111. Springer, 2009.

[466] The Common Criteria Recognition Agreement Members. Protection profiles for secure signature creation device - Part 2: Device with key generation (BSI-CC-PP-0059-2009). URL https://www.commoncriteriaportal.org/files/ppfiles/pp0059b_pdf.pdf [last accessed: Aug. 2017], 2009.

[467] The Common Criteria Recognition Agreement Members. Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model (Version 3.1 Revision 4). URL https://www.commoncriteriaportal.org/cc/ [last accessed: Jan. 2017], 2012.

[468] The Common Criteria Recognition Agreement Members. Protection Profile for Trusted Signature Creation Module in TW4S. URL https://www.commoncriteriaportal.org/files/ppfiles/pp-tscm-v1.2.pdf [last accessed: Jan. 2017], 2016.

[469] The Council of the European Union. Council Directive 76/207/EEC of 9 February 1976 on the implementation of the principle of equal treatment for men and women as regards access to employment, vocational training and promotion, and working conditions. *Official Journal* , OJ L 39 of 14.2.1976:40–42. 1976.

[470] The Council of the European Union. Council Directive 2010/45/EU of 13 July 2010 amending Directive 2006/112/EC on the common system of value added tax as regards the rules on invoicing. *Official Journal* , OJ L 189:1–8. 2010.

[471] The Government of the Hong Kong Special Administrative Region. An overview of information security standards. URL http://www.infosec.gov.hk/english/technical/files/overview.pdf [last accessed: Apr. 2015], 2008.

[472] The National Archives. Redaction toolkit – editing exempt information from paper and electronic documents prior to release. URL http://www.nationalarchives.gov.uk/documents/information-management/redaction_toolkit.pdf [last accessed: Aug. 2016], 2011.

[473] E. Z. Tragos, V. Angelakis, A. Fragkiadakis, D. Gundlegard, S. Nechifor, G. Oikonomou, H. C. Pöhls, and A. Gavras. Enabling reliable and secure iot-based smart city applications. In *Proc. of Conference on Pervasive Computing and Communications Workshops (IEEE PERCOM Workshops 2014)*, pages 111–116. IEEE, 2014. URL http://dx.doi.org/10.1109/PerComW.2014.6815175 [last accessed: Sep. 2017].

[474] E. Z. Tragos, H. C. Pöhls, R. C. Staudemeyer, D. Slamanig, A. Kapovits, S. Suppan, A. Fragkiadakis, G. Baldini, R. Neisse, P. Langendörfer, Z. Dyka, and C. Wittke. Securing the Internet of Things – Security and Privacy in a Hyperconnected World. In O. Vermesan and P. Friess, editors, *Building the Hyperconnected Society — IoT Research and Innovation Value Chains, Ecosystems and Markets*. RIVER PUBLISHERS, 2015. ISBN 978-87-93237-99-5.

[475] G. Traverso, D. Demirel, and J. A. Buchmann. *Homomorphic Signature Schemes - A Survey*. Springer Briefs in Computer Science. Springer, 2016. ISBN 978-3-319-32114-1. URL https://doi.org/10.1007/978-3-319-32115-8 [last accessed: Sep. 2017].

[476] United Kingdom Department for Business, Enterprise and Regulatory Reform. BERR guidance on electronic signatures, 2009. URL http://webarchive.nationalarchives.gov.uk/20090609003228/http://www.berr.gov.uk/files/file49952.pdf [last accessed: Sep. 2017].

[477] United Kingdom Food Standard Agency (FSA). Guidance on Identifying the Food Business Operator and Changes in Operator for FSA Approval Purposes. [online, last accesssed Feb. 2016], 2007. URL https://www.food.gov.uk/sites/default/files/multimedia/pdfs/enforcement/idingfbochangesoperatorguide.pdf [last accessed: Sep. 2017].

[478] United Kingdom Information Commissioner's Office (ICO). Freedom of Information Act - Awareness Guidance No 5 (V3.0). URL https://ico.org.uk/media/for-organisations/documents/1178/awareness_guidance_5_v3_07_03_08.pdf [last accessed: Jun. 2016], 2008.

[479] United Kingdom Information Commissioner's Office (ICO). Safeguarding national security (section 24) – version : 1.0. URL https://ico.org.uk/media/for-organisations/documents/1174/safeguarding_national_security_section_24_foi.pdf [last accessed: Aug. 2016], 2012.

[480] United Kingdom Information Commissioner's Office (ICO). How to disclose information safely – removing personal data from information requests and datasets. URL https://ico.org.uk/media/1432979/how-to-disclose-information-safely.pdf [last accessed: Aug. 2016], 2015.

[481] United Kingdom Ministry of Justice. Freedom of information guidance – Exemptions guidance Section 43 – Commercial interests. URL http://webarchive.nationalarchives.gov.uk/+/http:/www.justice.gov.uk/docs/foi-exemption-s43.pdf [last accessed: Oct. 2016], 2008.

[482] United Kingdom Ministry of Justice. Lord Chancellor's Code of Practice on the management of records issued under section 46 of the Freedom of Information Act 2000. URL http://www.nationalarchives.gov.uk/documents/foi-section-46-code-of-practice.pdf [last accessed: Jun. 2016], 2009.

[483] United Nations. United Nations Directories for Electronic Data Interchange for Administration, Commerce and Transport (UN/EDIFACT). URL http://www.unece.org/trade/untdid/d01b/trmd/desadv_c.htm [last accessed: Jan. 2017], 2001.

[484] United States Congress. The Health Insurance Portability and Accountability Act of 1996 (HIPAA). P.L. No. 104-191, 110 Stat. 1938, 1996.

[485] United States Congress. Health Information Technology for Economic and Clinical Health Act of 2009 (HITECH). P.L. No. 115-5, 123 STAT. 115, 2009.

[486] U.S. Department of Health & Human Services. 45 CFR Parts 160, 162, and 164 – Health Insurance Reform: Security Standards; Final Rule. Federal Register/Vol. 68, No. 34, https://www.gpo.gov/fdsys/pkg/FR-2003-02-20/pdf/03-3877.pdf [last accessed: Oct. 2016], 2003.

[487] U.S. Department of Health & Human Services. Guidance to Render Unsecured Protected Health Information Unusable, Unreadable, or Indecipherable to Unauthorized Individuals. URL https://www.hhs.gov/hipaa/for-professionals/breach-notification/guidance/index.html [last accessed: Dec. 2016], 2016.

[488] U.S. Department of Justice. Law 107-347. *The E-Government Act*. 2002.

[489] U.S. Department of Justice. Response to a FOIA reqest – CRM-DOC1. URL https://www.aclu.org/files/pdfs/email-content-foia/DOJCrimDivdocs/CRM-1.pdf [last accessed: Feb. 2017], 2003.

[490] U.S. Department of Justice. The Freedom of Information Act, 5 U.S.C. §552, As Amended By Public Law No. 110-175, 121 Stat. 2524, and Public Law No. 111-83, §564, 123 Stat. 2142, 2184. URL http://www.justice.gov/sites/default/files/oip/legacy/2014/07/23/amended-foia-redlined-2010.pdf [last accessed: Jan. 2017], 2010.

[491] U.S. Office for Civil Rights (OCR). Guidance Regarding Methods for De-identification of Protected Health Information in Accordance with the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule. URL https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/understanding/coveredentities/De-identification/hhs_deid_guidance.pdf [last accessed: Jan. 2017], 2012.

[492] M. Vanhoef and F. Piessens. Key reinstallation attacks: Forcing nonce reuse in WPA2. In *Proc. of ACM conference on Computer and Communications Security (CCS 2017)*, pages 1313–1328. ACM, 2017.

[493] O. Vermesan, P. Friess, G. Baldini, T. Peirce, M. Botterman, M. C. Talacchini, A. Pereira, M. Handte, D. Rotondi, H. C. Pöhls, O. Vermesan, A. Baddii, B. Copigneaux, D. Schreckling, L. Vigano, G. Steri, S. Piccione, P. Vlacheas, V. Stavroulaki, D. Kelaidonis, R. Neisse, E. Tragos, P. Smadja, C. Hennebert, M. Serrano, S. Severi, G. Abreu, P. T. Kirstein, S. Varakliotis, and A. Skarmeta. *Internet of Things – Governance, Privacy and Security Issues*. IERC, 2015. URL `http://www.internet-of-things-research.eu/pdf/IERC_Position_Paper_IoT_Governance_Privacy_Security_Final.pdf` [last accessed: Sep. 2017].

[494] W. Viefhues. Das Gesetz über die Verwendung elektronischer Kommunikationsformen in der Justiz. In *NJW Neue juristische Wochenzeitung*, page 1009 ff. C.H.BECK, 2005.

[495] D. von Oheimb. IT security architecture approaches for Smart Metering and Smart Grid. In *Proc. of International Workshop on Smart Grid Security (SmartGridSec 2012)*, volume 7823 of *LNCS*, pages 1–25. Springer, 2012. URL `https://doi.org/10.1007/978-3-642-38030-3_1` [last accessed: Sep. 2017].

[496] W3C. XML Schema, 2004. URL `www.w3.org/XML/Schema` [last accessed: Sep. 2017].

[497] G. Wang. Designated-verifier proxy signatures for e-commerce. In *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*, volume 3, pages 1731–1734. IEEE, 2004.

[498] Y. Wang, H. Pang, and R. H. Deng. Verifiably encrypted cascade-instantiable blank signatures to secure progressive decision management. *International Journal of Information Security*, pages 1–17. Springer, 2017. URL `http://dx.doi.org/10.1007/s10207-017-0372-2` [last accessed: Sep. 2017].

[499] Z. Wang. Improvement on Ahn et al.'s RSA P-Homomorphic Signature Scheme. In *Proc. of International Conference on Security and Privacy in Communication Networks (SecureComm 2012)*, volume 106 of *LNICST*, pages 19–28. Springer, 2013. URL `http://dx.doi.org/10.1007/978-3-642-36883-7_2` [last accessed: Sep. 2017].

[500] S. D. Warren and L. D. Brandeis. The Right to Privacy. *Harvard Law Review*, 193(4). 1890.

[501] L. Wei, S. E. Coull, and M. K. Reiter. Bounded vector signatures and their applications. In *Proc. of ACM Symposium on Information, Computer and Communications Security (ASIA CCS 2011)*, pages 277–285. ACM, 2011. URL `http://doi.acm.org/10.1145/1966913.1966949` [last accessed: Sep. 2017].

[502] S. Wettig and E. Zehendner. The electronic agent: a legal personality under german law. In *Proc. of the Law and Electronic Agents workshop (LEA'03)*, pages 97–112, 2003.

[503] Wolff. Beck'scher Online-Kommentar Datenschutzrecht. In *Beck'scher Online-Kommentar*. C.H.BECK, 3rd edition, 2013.

[504] Working Party on Facilitation of International Trade Procedures at the United Nations Economic Commission for Europe (UNECE). Recommendation No. 14: "Authentication of Trade Documents by Means other than Signature", 1979.

[505] World Health Organization (WHO). Informed consent form template for clinical studies. URL `http://www.who.int/rpc/research_ethics/informed_consent/en/` [last accessed: Oct. 2016], 2008.

[506] World Trade Organization (WTO). URUGUAY ROUND AGREEMENT: Trade-Related Aspects of Intellectual Property Rights The TRIPS Agreement is Annex 1C of the Marrakesh Agreement Establishing the World Trade Organization. URL `http://www.wto.org/english/docs_e/legal_e/27-trips.pdf`, 1994.

[507] Z.-Y. Wu, C.-W. Hsueh, C.-Y. Tsai, F. Lai, H.-C. Lee, and Y. Chung. Redactable Signatures for Signed CDA Documents. *Journal of Medical Systems*, 36(3):1795–1808. Springer, 2012. ISSN 0148-5598.

[508] T. H. Yuen, W. Susilo, J. K. Liu, and Y. Mu. Sanitizable signatures revisited. In *Proc. of Cryptology and Network Security (CANS 2008)*, volume 5339 of *LNCS*, pages 80–97. Springer, 2008. URL `http://dx.doi.org/10.1007/978-3-540-89641-8_6` [last accessed: Sep. 2017].

[509] D. H. Yum and P. J. Lee. Sanitizable Signatures Reconsidered. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E94-A(2):717–724. IEICE, 2011.

[510] D. H. Yum, J. W. Seo, and P. J. Lee. Trapdoor sanitizable signatures made easy. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2010)*, volume 6123 of *LNCS*, pages 53–68. Springer, 2010. ISBN 3-642-13707-5, 978-3-642-13707-5. URL `portal.acm.org/citation.cfm?id=1894302.1894308` [last accessed: Sep. 2017].

[511] F. Zhang, R. Safavi-naini, and W. Susilo. ID-Based Chameleon Hashes from Bilinear Pairings. *IACR Cryptology ePrint Archive*, 208. 2003. URL `https://eprint.iacr.org/2003/208.ps.gz` [last accessed: Sep. 2017].

[512] J. Zhou and D. Gollmann. An efficient non-repudiation protocol. In *Proc. of Computer Security Foundations Workshop (CSFW 1997)*, pages 126–132, 1997. URL `http://dx.doi.org/10.1109/CSFW.1997.596801` [last accessed: Sep. 2017].

[513] B. Zwattendorfer and D. Slamanig. On Privacy-Preserving Ways to Porting the Austrian eID System to the Public Cloud. In *Proc. of IFIP TC-11 International Information Security Conference (IFIP SEC 2013)*, pages 300–314. Springer, 2013. URL `https://doi.org/10.1007/978-3-642-39218-4_23` [last accessed: Sep. 2017].

[514] B. Zwattendorfer and D. Slamanig. Design strategies for a privacy-friendly Austrian eID system in the public cloud. *Journal of Computers & Security*, 52:178–193. Elsevier, 2015.

[515] B. Zwattendorfer and D. Slamanig. The Austrian eID Ecosystem in the Public Cloud: How to Obtain Privacy while Preserving Practicality. *Journal of Information Security and Applications*, 27:35–53. Elsevier, 2016.

A — Record of Publications

Overview of Appendix A

Selected results of this thesis have been published as peer-reviewed articles in conference proceedings and journals, or by other means like presentations or technical reports. A total of 22 peer-reviewed publications (including 2 journal articles) and 10 technical reports directly relate to the findings of this thesis; they contain early results or results with a selected focus. The related publications are grouped by their nature (conference or journal) and in each section ordered chronologically.

Appendix A.1 lists peer-reviewed articles in journals.

Appendix A.2 lists peer-reviewed publications in conference proceedings.

Links to the related section(s) within this thesis are additionally provided for the above listed publications. Additionally, this section lists further publications co-authored by the author of this thesis.

Appendix A.3 contains a list of other publications like technical and project reports that contain content authored by the author which is related to this thesis.

Appendix A.4 lists the remaining peer-reviewed publications that were co-authored by the author of this thesis in the time before submitting this work.

A.1. Peer-reviewed articles in journals related to this thesis

n° 1 **Rechtsfolgen editierbarer Signaturen**
[233] F. Höhne, H. C. Pöhls, and K. Samelin. Rechtsfolgen editierbarer Signaturen. *Datenschutz und Datensicherheit - DuD*, 36(7):485–491. Springer, 2012. URL `http://dx.doi.org/10.1007/s11623-012-0165-8` [last accessed: Sep. 2017]

Relates to:
- (Sec. 17.3) The publication disseminates the overall result that an MSS can generate a qualified digital signature when the needed properties are cryptographically maintained, but other than the full analysis of the thesis the published results are limited to the German legislative framework under the former Directive 1999/93/EC.

n° 2 **Redactable Signature Schemes for Trees with Signer-Controlled Non-Leaf-Redactions**
[134] H. de Meer, H. C. Pöhls, J. Posegga, and K. Samelin. Redactable Signature Schemes for Trees with Signer-Controlled Non-Leaf-Redactions. *Journal of E-Business and Telecommunications*, 455:155–171. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-662-44791-8_10` [last accessed: Sep. 2017]

Relates to:
- (Sec. 12.5) The article disseminates a solution to overcome the shortcoming of reduced flexibility when it comes to structural integrity protection, e.g., it allows to authorize the redaction of non-leaf nodes to flattening a sorted list into a set by removing structure;
- (Sec. 14.7 and Sec. 14.8) New scheme $flex\mathcal{RSS}$.

A.2. Peer-reviewed articles in conference proceedings related to this thesis

nº 3 **ConCert: Content Revocation using Certificates**

[376] H. C. Pöhls. ConCert: Content revocation using certificates. In *Proc. of the Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik (GI Sicherheit 2008)*, volume 128 of *GI-Edition Lecture Notes in Informatics (LNI)*, pages 149–162. GI, 2008. URL `http://henrich.poehls.com/papers/2008_poehls_GI-Sicherheit_Concert-Content-revocation-using-certificates.pdf` [last accessed: Sep. 2017]. URL `http://137.226.34.227/dblp/db/conf/sicherheit/sicherheit2008.html` [last accessed: Sep. 2017]

Relates to:
- (Sec. 1.2.1) The publication shares the same motivation: Underlying idea is that a digital signature could be used to signal (and verifiably capture) legally recognised consent (by authentication of origin) and technically protects integrity of the data blocks while allowing to keep unneeded blocks confidential.

nº 4 **Verifiable and Revocable Expression of Consent to Processing of Aggregated Personal Data**

[375] H. C. Pöhls. Verifiable and revocable expression of consent to processing of aggregated personal data. In *Proc. of International Conference on Information and Communications Security (ICICS 2008)*, volume 5308 of *LNCS*, pages 279–293. Springer, 2008. URL `http://dx.doi.org/10.1007/978-3-540-88625-9_19` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2008_Poehls_RevocableExpressionOfConsent_ICICS2008.pdf` [last accessed: Sep. 2017]

Relates to:
- (Sec. 1.2.1) The paper has the same underlying motivation: Design a legally recognised digital signatures scheme that allows to signal legal consent by authentication of origin and also technically protects integrity of the data's blocks while allowing to keep unneeded blocks confidential.
- (Sec. 14.4) This early publication already states the basic idea that became generalised in this thesis to the concept of merging (in this early paper called aggregation) and redaction (in this early paper called splitting) of signed blocks of data.
- (Sec. 18.5) The paper contains the application scenario of MSS to control the flow of personal data and represent consent.

nº 5 **Towards Automated Processing of the Right of Access in Inter-organizational Web Service Compositions**

[226] R. Herkenhöner, H. de Meer, M. Jensen, and H. C. Pöhls. Towards Automated Processing of the Right of Access in Inter-organizational Web Service Compositions. In *Proc. of IEEE World Congress on Services (SERVICES 2010)*, pages 645–652. IEEE Computer Society, 2010. ISBN 978-0-7695-4129-7. URL `http://dx.doi.org/10.1109/SERVICES.2010.56` [last accessed: Sep. 2017]

Relates to:
- (Sec. 18.5) The paper contains the application scenario: Exercising data governance rights, e.g., your right of information, in layered service orientated architectures.

nº 6 **Sanitizable Signatures in XML Signature — Performance, Mixing Properties, and Revisiting the Property of Transparency**

[391] H. C. Pöhls, K. Samelin, and J. Posegga. Sanitizable Signatures in XML Signature - Performance, Mixing Properties, and Revisiting the Property of Transparency. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2011)*, volume 6715 of *LNCS*, pages 166–182. Springer, 2011. URL `http://dx.doi.org/10.1007/978-3-642-21554-4_10` [last accessed: Sep. 2017]

Relates to:
- (Sec. 3.10.1) The paper contains the basic argument of the high-level definition of transparency found in this thesis;
- (Sec. 6.3.2) The paper's weak and strong transparency notions are the basis for the extension of Integrity with aspect of future change detection;
- (Sec. 6.5) Analysis of *Ateniese et al.*'s transparency properties is taken further in this thesis; here only the basic idea of the separation of block- and message-level properties is captured;

- (Sec. 11.6.4) The problems described in this paper gave rise to the refinement and harmonisation of the MSS property transparency done in this thesis.

n° 7 **The Role of Data Integrity in EU Digital Signature Legislation — Achieving Statutory Trust for Sanitizable Signature Schemes**

[381] H. C. Pöhls and F. Höhne. The Role of Data Integrity in EU Digital Signature Legislation - Achieving Statutory Trust for Sanitizable Signature Schemes. In *Revised Selected Papers from the 7th International Workshop on Security and Trust Management (STM 2011)*, volume 7170 of *LNCS*, pages 175–192. Springer, 2011. URL `http://dx.doi.org/10.1007/978-3-642-29963-6_13` [last accessed: Sep. 2017]

Relates to:

- (Sec. 4.6.4) The publication contains mentions the legal analysis result that any subsequent modification must be detected;
- (Chapter 5) The publication contains some of the existing legal integrity definitions analysed; (Subsections 4.3.2 to 4.3.4 and 4.4.4)
- (Chapter 5) The publication contains some of the existing technical integrity definitions analysed; (Subsections 5.1.2, 5.1.3 and 5.1.7 to 5.1.9)
- (Chapter 6) The paper gives an initial concept of two new aspects in the Integrity definition: Allowed changes (*CA) and detection (*FD and *CD); the thesis contains the completed version with an additional third aspect;
- (Sec. 17.3.3) The paper distributed the legal analysis that yields an MSS that offers at least ACA – 1CD integrity protection can be argued to be legally equivalent to advanced electronic signatures following Directive 1999/93/EC as it can technically achieve the same technical protection as CDSS for unmodified documents.

n° 8 **Sanitizable Signed Privacy Preferences for Social Networks**

[390] H. C. Pöhls, A. Bilzhause, K. Samelin, and J. Posegga. Sanitizable Signed Privacy Preferences for Social Networks. In *Proc. of the GI Workshop on Privacy and Identity Management for Communities - Communities for Privacy and Identity Management (DICCDI 2011)* , volume 192 of *GI-Edition Lecture Notes in Informatics (LNI)*. GI, 2011. URL `http://www.user.tu-berlin.de/komm/CD/paper/090212.pdf` [last accessed: Sep. 2017]

Relates to:

- (Sec. 14.4) The paper informally describes the concept of merging (in this paper referred to as “Reconstructability”) and its impact in the application domain of personal data;
- (Sec. 18.5) The paper describes the application of RSS for personal data management; it describes how to facilitate the data subject’s redactable signature to retain a verifiable consent.

n° 9 **Sticky Signatures: Legal Advantages of Redactable Signatures and Credentials in the Food Supply Chain**

[382] H. C. Pöhls and F. Höhne. Sticky Signatures: Legal Advantages of Redactable Signatures and Credentials in the Food Supply Chain. In *Proc. of Interdisciplinary Conference on Current Issues in IT Security 2012*, Interdisciplinary Series of the Max Planck Institute for Foreign and International Criminal Law (MPICC). Dunker & Humblot, Berlin, 2012

Relates to:

- (Sec. 1.2.3) The publication contains an example used in the legal motivation for high probative value and applications of RSS;
- (Sec. 8.4.3 and Sec. 8.4.4) The publication gives an example that legal data protection as well as trade-secret requires privacy to be strong to not make sure that redacted blocks cannot be reconstructed without knowledge of the original;
- (Sec. 7.8 and Sec. 18.4) The publication gives an example for the legal value of the property of linkability.

nº 10 **On Structural Signatures for Tree Structured Data**

[425] K. Samelin, H. C. Pöhls, A. Bilzhause, J. Posegga, and H. de Meer. On Structural Signatures for Tree Data Structures. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2012)*, volume 7341 of *LNCS*, pages 171–187. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-31284-7_11` [last accessed: Sep. 2017]

Relates to:

- (Sec. 14.5 and Sec. 14.6) The paper contains the scheme $priv\mathcal{RSS}$: extension of the security model to allow non-leaf redactions and still have the Signer in control of authorized level-promotion and root-redactability (Signer must allow redaction of parent or intermediate nodes);
- (Sec. 12.5) The paper showed the attack on existing schemes' structural integrity termed 'level-promotion';
- (Sec. 12.1) The paper also showed the attacks on privacy and transparency of existing schemes.

nº 11 **Non-Interactive Public Accountability for Sanitizable Signatures**

[68] C. Brzuska, H. C. Pöhls, and K. Samelin. Non-Interactive Public Accountability for Sanitizable Signatures. In *Revised Selected Papers of European PKI Workshop: Research and Applications (EuroPKI 2012)*, volume 7868 of *LNCS*, pages 178–193. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-40012-4_12` [last accessed: Sep. 2017]

Relates to:

- (Sec. 6.4.3) The publication covers the basis of the definition of the non-interactive public accountability;
- (Sec. 13.1) The publication covers the formal definition of the non-interactive public accountability on the scope of message and blocks for SSS;
- (Sec. 13.4.1 and Sec. 13.5) The paper contains a brief description and proofs of the scheme $pubacc\mathcal{SSS}$.

nº 12 **Flexible Redactable Signature Schemes for Trees — Extended Security Model and Construction**

[394] H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Flexible Redactable Signature Schemes for Trees — Extended Security Model and Construction. In *Proc. of the International Conference on Security and Cryptography (SECRYPT 2012)*, pages 113–125. SciTePress, 2012. URL `http://t1p.de/SECRYPT2012` [last accessed: Sep. 2017]

Relates to:

- (Sec. 12.5) The paper contains the basics of the analysis of shortcoming of reduced flexibility when it comes to redaction of structure, including examples;
- (Sec. 14.7 and Sec. 14.8) The paper contains a brief description and proofs of the scheme $flex\mathcal{RSS}$: adds flexibility and expressiveness when defining authorized modifications for RSS on tree-structured data.

nº 13 **Redactable Signatures for Independent Removal of Structure and Content**

[426] K. Samelin, H. C. Pöhls, A. Bilzhause, J. Posegga, and H. de Meer. Redactable Signatures for Independent Removal of Structure and Content. In *Proc. of International Conference on Information Security Practice and Experience (ISPEC 2012)*, volume 7232 of *LNCS*, pages 17–33. Springer, 2012. URL `http://dx.doi.org/10.1007/978-3-642-29101-2_2` [last accessed: Sep. 2017]

Relates to:

- (Sec. 12.5) The paper gives the basic ideas for the analysis of shortcoming of reduced flexibility when it comes to redaction of structure, e.g. flattening a sorted list into a set by removing structure;
- (Sec. 14.9 and Sec. 14.10) The paper contains a brief description and proofs of the scheme $struct\mathcal{RSS}$: adds the possibility to define authorized modifications of structure also standalone in an RSS on tree-structured data;
- (Sec. 14.5.1.1) The paper disseminated the definition of the property of consecutive redaction control.

nº 14 **Contingency Revisited: Secure Construction and Legal Implications of Verifiably Weak Integrity**

[379] H. C. Pöhls. Contingency Revisited: Secure Construction and Legal Implications of Verifiably Weak Integrity. In *Proc. of IFIP WG 11.11 International Conference on Trust Management (IFIPTM 2013)*, volume 401 of *IFIPAICT*, pages 136 – 150. Springer, 2013. URL `http://dx.doi.org/10.1007/978-3-642-38323-6_10` [last accessed: Sep. 2017]

Relates to:
- (Chapter 6) The publication describes in detail the corner case of the integrity model of this thesis with ACA (see Sec. 6.2.3) termed Contingency; it shows that the model of this thesis is very versatile as it allows for the description of the "Dual of Integrity" [421] introduced by *Rost and Pfitzmann* and named "contingency" [33] by *Bedner and Ackermann* as a corner case of sanitizable signatures;
- (Sec. 18.8) The publication is one example for the wide range of applicability of MSS; with contingency being a special case of the legal impact of a 3^{rd}-party verifiability of the consented delegation of signing rights.

nº 15 **Malleable Signatures for Resource Constrained Platforms**

[395] H. C. Pöhls, S. Peters, K. Samelin, J. Posegga, and H. de Meer. Malleable Signatures for Resource Constrained Platforms. In *Proc. of IFIP WG 11.2 International Workshop in Information Security Theory and Practice (WISTP 2013)*, number 7886 in LNCS, pages 18–33. Springer, 2013. URL `https://dx.doi.org/10.1007/978-3-642-38530-8_2` [last accessed: Sep. 2017]

Relates to:
- (Sec. 13.10) The paper exhibits the proof-of-concept implementation of several SSS running inside a smart card, e.g. the scheme from [68];
- (Sec. 14.15) The paper shows the proof-of-concept implementation of an RSS on a smart card.

nº 16 **Scope of Security Properties of Sanitizable Signatures Revisited**

[132] H. de Meer, H. C. Pöhls, J. Posegga, and K. Samelin. Scope of Security Properties of Sanitizable Signatures Revisited. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2013)*, pages 188–197. IEEE, 2013. URL `http://dx.doi.org/10.1109/ARES.2013.26` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2013_DeMeer_Poehls_Posegga_Samelin-ScopeofSecurityPropertiesofSanitizableSignaturesRevisited.pdf` [last accessed: Sep. 2017]

Relates to:
- (Sec. 13.2) The paper contains the new property of Group-of-Blocks-level Non-Interactive Public Accountability for SSS;
- (Sec. 13.6 and Sec. 13.7) The paper contains a brief description and proofs of the scheme $blockacc\mathcal{SSS}$ that achieves the properties of accountability and transparency at the scope of blocks and groups of blocks instead of being of a message-wide scope.

nº 17 **Efficient and Perfectly Unlinkable Sanitizable Signatures without Group Signatures**

[69] C. Brzuska, H. C. Pöhls, and K. Samelin. Efficient and Perfectly Unlinkable Sanitizable Signatures without Group Signatures. In *Revised Selected Papers of 10th European PKI Workshop: Research and Applications (EuroPKI 2013)*, volume 8341 of *LNCS*, pages 12–30. Springer, 2013. URL `http://dx.doi.org/10.1007/978-3-642-53997-8_2` [last accessed: Sep. 2017]

Relates to:
- (Sec. 13.3) The paper contains the description of the new strengthened property of unlinkability;
- (Sec. 13.8 and Sec. 13.9) The paper contains a brief description and proof of the scheme $unlinkable\mathcal{SSS}$.

nº 18 **Redactable Signatures to Control the Maximum Noise for Differential Privacy in the Smart Grid**

[383] H. C. Pöhls and M. Karwe. Redactable signatures to control the maximum noise for differential privacy in the smart grid. In *Proc. of International Workshop on Smart Grid Security (SmartGridSec 2014)*, volume 8448 of *LNCS*, pages 79–93. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-319-10329-7_6` [last accessed: Sep. 2017]

Relates to: – (Sec. 18.3) The publication shows the applicability of a private RSS that enables to delegate the content removal for increased data protection to a partly trusted party while still retaining a legally valuable signature. This showcases the need for achieving both: the cryptographic property of privacy for data protection and retaining a high probative value for the data originating from a trusted signer.

nº 19 **On the Relation between Redactable and Sanitizable Signature Schemes**

[133] H. de Meer, H. C. Pöhls, J. Posegga, and K. Samelin. On the Relation between Redactable and Sanitizable Signature Schemes. In *Proc. of International Symposium on Engineering Secure Software and Systems (ESSoS 2014)*, volume 8364 of *LNCS*, pages 113–130. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-319-04897-0_8` [last accessed: Sep. 2017]

Relates to: – (Chapter 15) This publication encapsulates the thesis underlying argument that SSS and RSS are two distinct concepts, which cannot be emulated without loosing efficiency or security.

nº 20 **On Updatable Redactable Signatures**

[387] H. C. Pöhls and K. Samelin. On updatable redactable signatures. In *Proc. of International Conference on Applied Cryptography and Network Security (ACNS 2014)*, volume 8479 of *LNCS*, pages 457–475. Springer, 2014. URL `http://dx.doi.org/10.1007/978-3-319-07536-5_27` [last accessed: Sep. 2017]

Relates to:

- (Sec. 14.11 and Sec. 14.12) The paper contains a brief description and proof of the scheme $merge\mathcal{RSS}$;
- (Sec. 12.6) The paper contains the shortcoming of no explicit inverse operation for redact in existing schemes;
- (Sec. 14.4) The paper exhibits the definition of mergeability and related properties.

nº 21 **Accountable Redactable Signatures**

[388] H. C. Pöhls and K. Samelin. Accountable Redactable Signatures. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2015)*, pages 60 – 69. IEEE, 2015. URL `http://dx.doi.org/10.1109/ARES.2015.10` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2015_Poehls-Samelin_Accountable-Redactable-Signatures_ARES15.pdf` [last accessed: Sep. 2017]

Relates to:

- (Sec. 14.13 and Sec. 14.14) The paper contains a brief description and proof of the scheme $pubacc\mathcal{RSS}$ allowing an RSS to gain the accountability property from an accountable SSS;
- (Sec. 14.1) The publication exhibits the description of the new RSS properties related to accountability;
- (Sec. 17.3.4) The publication disseminates the result that an RSS has the ability to achieve the public form of accountability (PUB), this is one basis for the argumentation that RSS can achieve a probative value equal to standard digital signatures.

nº 22 **The legal status of malleable- and functional signatures in light of Regulation (EU) No 910/2014**

[199] F.W.J. van Geelkerken, H. C. Pöhls, and S. Fischer-Hübner. The legal status of malleable- and functional signatures in light of Regulation (EU) No 910/2014. In *Proc. of International Academic Conference of Young Scientists on Law & Psychology 2015 (LPS 2015)*, pages 404–410. L'viv Polytechnic Publishing House, 2015. ISBN 978-617-607-856-2. URL `https://drive.google.com/file/d/0B-Yu3Ni9z3PXM2lBajhCXzhoWk0/view` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2015_GeelkerkenPoehlsHuebner_legal_status_of_malleable_and_functional_signatures_in_eidas.pdf` [last accessed: Sep. 2017]

Relates to: – (Sec. 17.3) The publication disseminates the general result that an MSS can generate a qualified digital signature in accordance with Regulation 910/2014 if the needed properties are met, but the full analysis of the thesis goes beyond it and offers unpublished details.

A.3. Technical reports and project deliverables related to this thesis

For a timely dissemination of his results the author of this thesis used technical reports to publish his results, some of them later where converted into peer reviewed publications listed above. Further, the author has had the honour of working on topics related to his thesis in the projects ReSCUeIT [400], RERUM [221] and PRISMACLOUD [137]. Hence, the author is the collaborative author or editor of several project deliverables that were released during the time of preparing this thesis. Not surprisingly, a lot of the author's own material and publications influenced and appeared in those projects' reports. The following list provides an informative overview of which technical reports or project deliverables are related:

nº 23 **Authenticity and Revocation of Web Content using Signed Microformats and PKI**
[374] H. C. Pöhls. Authenticity and Revocation of Web Content using Signed Microformats and PKI. Technical Report B-276-07, University of Hamburg, Department of Informatics, Hamburg, Germany, 2007. URL `http://edoc.sub.uni-hamburg.de/informatik/volltexte/2009/16/pdf/B_276_07.pdf` [last accessed: Sep. 2017]

nº 24 **Length-Hiding Redactable Signatures from One-Way Accumulators in $O(n)$**
[392] H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Length-Hiding Redactable Signatures from One-Way Accumulators in $\mathcal{O}(n)$ (MIP-1201). Technical Report MIP-1201, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP-1201.pdf` [last accessed: Sep. 2017]

nº 25 **Transparent Mergeable Redactable Signatures with Signer Commitment and Applications**
[393] H. C. Pöhls, K. Samelin, J. Posegga, and H. de Meer. Transparent Mergeable Redactable Signatures with Signer Commitment and Applications (MIP-1206). Technical Report MIP-1206, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/TR_MIP1206.pdf` [last accessed: Sep. 2017]

nº 26 **Redactable vs. Sanitizable Signatures**
[427] K. Samelin, H. C. Pöhls, J. Posegga, and H. de Meer. Redactable vs. Sanitizable Signatures (MIP-1208). Technical Report MIP-1208, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP-1208_neu.pdf` [last accessed: Sep. 2017]

nº 27 **Block-level Accountability for Transparent Sanitizable Signatures**
[428] K. Samelin, H. C. Pöhls, J. Posegga, and H. de Meer. Block-level Accountability for Transparent Sanitizable Signatures (MIP-1209). Technical report, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `http://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP1209.pdf` [last accessed: Sep. 2017]

nº 28 **Indistinguishability of One-Way Accumulators**
[131] H. de Meer, M. Liedel, H. C. Pöhls, J. Posegga, and K. Samelin. Indistinguishability of One-Way Accumulators (MIP-1210). Technical Report MIP-1210, Faculty of Computer Science and Mathematics (FIM), University of Passau, 2012. URL `https://www.fim.uni-passau.de/fileadmin/files/forschung/mip-berichte/MIP_1210.pdf` [last accessed: Sep. 2017]

nº 29 **RERUM D3.1: Enhancing the autonomous smart objects and the overall system security of IoT based Smart Cities** (related are Sec. 2.3.3–2.3.9)
[423] D. Ruiz and E. al. RERUM Deliverable D3.1 - Enhancing the autonomous smart objects and the overall system security of IoT based Smart Cities, 2015. URL `https://bscw.ict-rerum.eu/pub/bscw.cgi/d14540/RERUMdeliverableD3_1.pdf` [last accessed: Sep. 2017]

nº 30 **RERUM D3.2: Privacy enhancing techniques in the Smart City applications** (related are Sec. 3.9 and Sec. 4.2–4.3)
[221] H. C. Pöhls et al. RERUM Deliverable D3.2 - Privacy enhancing techniques in the Smart City applications, 2015. URL `http://cordis.europa.eu/docs/projects/cnect/4/609094/080/deliverables/001-RERUMdeliverableD32Ares20153669911.pdf` [last accessed: Sep. 2017]

nº 31 **PRISMACLOUD D2.1: Legal, Social and HCI Requirements** (related is Sec. 5)
[9] A. Alaqra, S. Fischer-Hübner, J. S. Pettersson, F. van Geelkerken, E. Wästlund, M. Volkamer, T. Länger, and H. C. Pöhls. PRISMACLOUD public deliverable D2.1: Legal, Social and HCI Requirements. URL `https://prismacloud.eu/wp-content/uploads/2017/03/D2.1-Legal-social-and-HCI-requirements_v1.2.pdf` [last accessed: Sep. 2017], 2015

nº 32 **PRISMACLOUD D4.4: Overview of Functional and Malleable Signature Schemes** (related is Sec. 5)
[137] D. Demirel, D. Derler, C. Hanser, H. C. Pöhls, D. Slamanig, and G. Traverso. PRISMACLOUD public deliverable D4.4: Overview of Functional and Malleable Signature Schemes, 2016. URL `https://online.tugraz.at/tug_online/voe_main2.getvolltext?pCurrPk=86456` [last accessed: Sep. 2017]

A.4. Other peer-reviewed publications

The author of this thesis is the collaborative author of several other peer-reviewed publications. For the sake of completeness the following list provides the remaining peer-reviewed publications (co-)authored and published (or already accepted for publication) in the time before the submission of this thesis.

nº 33 **Smartcard Firewalls Revisited**
[386] H. C. Pöhls and J. Posegga. Smartcard firewalls revisited. In *Proc. of IFIP WG 8.8/11.2 International Conference on Smart Card Research and Advanced Applications (CARDIS 2006)*, volume 3928 of *LNCS*, pages 179 – 191. Springer, 2006. URL `http://www.springerlink.com/content/t42644k876vg612k/` [last accessed: Sep. 2017]

nº 34 **Die "Untiefen" der neuen XML-basierten Dokumentenformate**
[389] H. C. Pöhls and L. Westphal. Die ”Untiefen” der neuen XML-basierten Dokumentenformate. In *Proc. of DFN CERT Workshop Sicherheit in vernetztzen Systemen*, 2008. URL `https://www.dfn-cert.de/dokumente/workshop/2008/poehls.pdf` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2008_Poehls_Westphal_2008_DFN-CERT-WS_Untiefen_der_XML-Dokumentenformate.pdf` [last accessed: Sep. 2017]

nº 35 **Why Showing one TLS Certificate is not enough? Towards a Browser Feedback for Multiple TLS Certificate Verifications.**
[377] H. C. Pöhls. Why showing one TLS certificate is not enough? towards a browser feedback for multiple TLS certificate verifications. In *Proc. of the Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik (GI Sicherheit 2010)*, volume 170 of *GI-Edition Lecture Notes in Informatics (LNI)*, pages 265–276. GI, 2010. URL `http://henrich.poehls.com/papers/2010_Poehls_Show_Multiple_SSL_Certificate_Verifications_GI-Sicherheit.pdf` [last accessed: Sep. 2017]. `https://subs.emis.de/LNI/Proceedings/Proceedings170/265.pdf`

nº 36 **Staatliche Schutzpflichten für die IT-Infrastruktur**
[231] F. Höhne and H. C. Pöhls. Staatliche Schutzpflichten für die IT-Infrastruktur. In P. Schartner and E. Weippl, editors, *Proc. of D-A-CH Security 2010*. syssec, 2010. ISBN 978-3-00-031441-4

nº 37 **Grund und Grenzen staatlicher Schutzpflichten für die IT-Infrastruktur**
[232] F. Höhne and H. C. Pöhls. Grund und Grenzen staatlicher Schutzpflichten für die IT-Infrastruktur. In J. Taeger, editor, *Tagungsband der 11. Herbstakademie der Deutschen Stiftung für Recht und Informatik (DSRI): Digitale Evolution - Herausforderungen für das Informations- und Medienrecht*. OlWIR Oldenburger Verlag für Wirtschaft, Informatik und Recht, 2010. ISBN 978-3-939704-50-8

nº 38 **RERUM: Building a Reliable IoT upon Privacy- and Security-enabled Smart Objects**
[396] H. C. Pöhls, V. Angelakis, S. Suppan, K. Fischer, G. Oikonomou, E. Z. Tragos, R. D. Rodriguez, and T. Mouroutis. RERUM: Building a Reliable IoT upon Privacy- and Security- enabled Smart Objects. In *Proc. of the Workshop on Internet of Things Communications and Technologies (IEEE WCNC 2014)*, pages 122–127. IEEE, 2014. URL `http://dx.doi.org/10.1109/WCNCW.2014.6934872` [last accessed: Sep. 2017]

nº 39 **Enabling Reliable and Secure IoT-based Smart City Applications**
[473] E. Z. Tragos, V. Angelakis, A. Fragkiadakis, D. Gundlegard, S. Nechifor, G. Oikonomou, H. C. Pöhls, and A. Gavras. Enabling reliable and secure iot-based smart city applications. In *Proc. of Conference on Pervasive Computing and Communications Workshops (IEEE PERCOM Workshops 2014)*, pages 111–116. IEEE, 2014. URL `http://dx.doi.org/10.1109/PerComW.2014.6815175` [last accessed: Sep. 2017]

nº 40 **Aggregation and Perturbation in Practice: Case-Study of Privacy, Accuracy and Performance**

[397] H. C. Pöhls, B. Petschkuhn, J. Rückert, and M. Mössinger. Aggregation and perturbation in practice: Case-study of privacy, accuracy and performance. In *IEEE International Workshop on Computer-Aided Modeling Analysis and Design of Communication Links and Networks (IEEE CAMAD 2014)*, pages 183–187. IEEE, 2014. URL `http://dx.doi.org/10.1109/CAMAD.2014.7033231` [last accessed: Sep. 2017]

nº 41 **Building the Hyperconnected Society - IoT Research and Innovation Value Chains, Ecosystems and Markets**

[474] E. Z. Tragos, H. C. Pöhls, R. C. Staudemeyer, D. Slamanig, A. Kapovits, S. Suppan, A. Fragkiadakis, G. Baldini, R. Neisse, P. Langendörfer, Z. Dyka, and C. Wittke. Securing the Internet of Things – Security and Privacy in a Hyperconnected World. In O. Vermesan and P. Friess, editors, *Building the Hyperconnected Society — IoT Research and Innovation Value Chains, Ecosystems and Markets*. RIVER PUBLISHERS, 2015. ISBN 978-87-93237-99-5

nº 42 **Internet of Things – Governance, Privacy and Security Issues**

[493] O. Vermesan, P. Friess, G. Baldini, T. Peirce, M. Botterman, M. C. Talacchini, A. Pereira, M. Handte, D. Rotondi, H. C. Pöhls, O. Vermesan, A. Baddii, B. Copigneaux, D. Schreckling, L. Vigano, G. Steri, S. Piccione, P. Vlacheas, V. Stavroulaki, D. Kelaidonis, R. Neisse, E. Tragos, P. Smadja, C. Hennebert, M. Serrano, S. Severi, G. Abreu, P. T. Kirstein, S. Varakliotis, and A. Skarmeta. *Internet of Things – Governance, Privacy and Security Issues*. IERC, 2015. URL `http://www.internet-of-things-research.eu/pdf/IERC_Position_Paper_IoT_Governance_Privacy_Security_Final.pdf` [last accessed: Sep. 2017]

nº 43 **A General Framework for Redactable Signatures and New Constructions**

[143] D. Derler, H. C. Pöhls, K. Samelin, and D. Slamanig. A General Framework for Redactable Signatures and New Constructions. In *Proc. of International Conference on Information Security and Cryptology (ICISC 2015)*, volume 9558 of *LNCS*, pages 3–19. Springer, 2015. URL `https://link.springer.com/chapter/10.1007/978-3-319-30840-1_1` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2015_DerlerPoehlsSamelinSlamanig-GeneralFrameworkForRedactableSignatures_ICISC.pdf` [last accessed: Sep. 2017]

nº 44 **Towards a New Paradigm for Privacy and Security in Cloud Services**

[321] T. Lorünser, C. B. Rodriguez, D. Demirel, S. Fischer-Hübner, T. Gross, T. Länger, M. des Noes, H. C. Pöhls, B. Rozenberg, and D. Slamanig. Towards a New Paradigm for Privacy and Security in Cloud Services. In *Proc. of Cyber Security and Privacy EU Forum (CSP Forum 2015)*, volume 530 of *CCIS*, pages 14–25. Springer, 2015. URL `https://doi.org/10.1007/978-3-319-25360-2_2` [last accessed: Sep. 2017]

nº 45 **JSON Sensor Signatures (JSS): End-to-End Integrity Protection from Constrained Device to IoT Application**

[380] H. C. Pöhls. JSON Sensor Signatures (JSS): End-to-End Integrity Protection from Constrained Device to IoT Application. In *Proc. of the Workshop on Extending Seamlessly to the Internet of Things (esIoT) at the IMIS-2012 International Conference (IMIS 2015)*, pages 306 – 312. IEEE, 2015. URL `http://henrich.poehls.com/papers/2015_Poehls-JSONSensorSignatures_esIoT.pdf` [last accessed: Sep. 2017]. URL `http://dx.doi.org/10.1109/IMIS.2015.48` [last accessed: Sep. 2017]

nº 46 **Towards quantifying the cost of a secure IoT: Overhead and energy consumption of ECC signatures on an ARM-based device**

[353] M. Mössinger, B. Petschkuhn, J. Bauer, R. C. Staudemeyer, M. Wojcik, and H. C. Pöhls. Towards quantifying the cost of a secure IoT: Overhead and energy consumption of ECC signatures on an ARM-based device. In *Proc. of workshop on IoT-SoS: Internet of Things Smart Objects and Services (WOWMOM SOS-IOT 2016)*, pages 1–6. IEEE, 2016. URL `http://henrich.poehls.com/papers/2016_Moessinger_et_al-Towards_quantifying_the_cost_of_a_secure_IoT.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1109/WoWMoM.2016.7523559` [last accessed: Sep. 2017]

nº 47 **Selected Cloud Security Patterns to Improve End User Security and Privacy in Public Clouds**

[307] T. Länger, H. C. Pöhls, and S. Ghernaouti. Selected Cloud Security Patterns to Improve End User Security and Privacy in Public Clouds. In *Proc. of Annual Privacy Forum (APF 2016)*, volume 9857 of *LNCS*, pages 115–132. Springer, 2016. URL `http://henrich.poehls.com/papers/2016_LaengerPoehlsGhernaouti_SecurityPatternsForClouds_ENISA-APF.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1007/978-3-319-44760-5_8` [last accessed: Sep. 2017]

nº 48 PRISMACLOUD Tools: A Cryptographic Toolbox for Increasing Security in Cloud Services

[322] T. Lorünser, D. Slamanig, T. Länger, and H. C. Pöhls. PRISMACLOUD Tools: A Cryptographic Toolbox for Increasing Security in Cloud Services. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2016)*, pages 733–741, 2016. URL `http://henrich.poehls.com/papers/2016_LoruenserSlamanigLaengerPoehls_PRISMCLOUD-Architecture_SECPID_ARES.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1109/ARES.2016.62` [last accessed: Sep. 2017]

nº 49 Cryptographically Enforced Four-Eyes Principle

[48] A. Bilzhause, M. Huber, H. C. Pöhls, and K. Samelin. Cryptographically Enforced Four-Eyes Principle. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2016)*, pages 760–767. Conference Publishing Services (CPS), 2016. URL `http://henrich.poehls.com/papers/2016_BilzhauseHuberPoehlsSamelin_4EyesPrinciple_ARES_SECPID.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1109/ARES.2016.28` [last accessed: Sep. 2017]

nº 50 A Privacy Engineering Framework for the Internet of Things

[301] A. Kung, F. Kargl, S. Suppan, J. Cuellar, H. C. Pöhls, A. Kapovits, N. Notario, and Y. S. Martin. A Privacy Engineering Framework for the Internet of Things. In *Proc. of International Conference on Computers, Privacy and Data Protection 2016 (CDPD 2016)*, volume 36 of *LGTS*. Springer, 2016. URL `https://doi.org/10.1007/978-3-319-50796-5` [last accessed: Sep. 2017]

nº 51 Towards Authenticity and Privacy Preserving Accountable Workflows

[144] D. Derler, C. Hanser, H. C. Pöhls, and D. Slamanig. Towards Authenticity and Privacy Preserving Accountable Workflows. In *Proc. of IFIP Summer School on Privacy and Identity Management*, volume 476 of *IFIPAICT*, pages 170–186. Springer, 2016. URL `https://doi.org/10.1007/978-3-319-41763-9_12` [last accessed: Sep. 2017]. `http://henrich.poehls.com/papers/2015_DerlerHanserPoehlsSlamanig_Towards_Auth_Private_Accountable_Workflows.pdf` [last accessed: Sep. 2017]

nº 52 ECDSA on things: IoT integrity protection in practise

[29] J. Bauer, R. C. Staudemeyer, H. C. Pöhls, and A. Fragkiadakis. ECDSA on things: IoT integrity protection in practise. In *Proc. of Information and Communications Security (ICICS 2016)*, volume 9977 of *LNCS*, pages 3–17. Springer, 2016. URL `http://henrich.poehls.com/papers/2016_Bauer-et-al_ECDSA-on-things_ICICS.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1007/978-3-319-50011-9_1` [last accessed: Sep. 2017]

nº 53 Integrity and Authenticity Protection with Selective Disclosure Control in the Cloud and IoT

[198] C. Frädrich, H. C. Pöhls, W. Popp, N. Rakotondravony, and K. Samelin. Integrity and Authenticity Protection with Selective Disclosure Control in the Cloud & IoT. In *Proc. of Information and Communications Security (ICICS 2016)*, volume 9977 of *LNCS*, pages 197–213. Springer, 2016. URL `http://henrich.poehls.com/papers/2016_SelectiveDisclosureControl_ICICS2016_full.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1007/978-3-319-50011-9_16` [last accessed: Sep. 2017]

nº 54 An IoT middleware for enhanced security and privacy: the RERUM approach

[350] G. Moldovan, E. Z. Tragos, A. Fragkiadakis, H. C. Pöhls, and D. Calvo. An IoT Middleware for Enhanced Security and Privacy: The RERUM Approach. In *Proc. of IFIP International Conference on New Technologies, Mobility and Security (NTMS 2016)*, pages 1–5. IEEE, 2016. URL `http://henrich.poehls.com/papers/2016_Moldovan-et-al_IoT-middleware-for-enhanced-sec-and-priv-RERUM_NTMS.pdf` [last accessed: Sep. 2017]. URL `https://doi.org/10.1109/NTMS.2016.7792434` [last accessed: Sep. 2017]

nº 55 Chameleon-Hashes with Ephemeral Trapdoors And Applications to Invisible Sanitizable Signatures

[107] J. Camenisch, D. Derler, S. Krenn, H. C. Pöhls, K. Samelin, and D. Slamanig. Chameleon-Hashes with Ephemeral Trapdoors And Applications to Invisible Sanitizable Signatures (full version). In *Proc. of Public-Key Cryptography (PKC 2017)*, volume 10175 of *LNCS*, pages 152–182. Springer, 2017. URL `https://eprint.iacr.org/2017/011` [last accessed: Sep. 2017]. Full Version: Cryptology ePrint Archive, Report 2017/11

nº 56 **Einsetzbare Kryptografie für die Cloud**

[384] H. C. Pöhls and T. Länger. Einsetzbare Kryptografie für die Cloud. *Zeitschrift für Datenrecht und Informationssicherheit (digma)*, 17(1):78–81. Schulthess Juristische Medien, 2017

nº 57 **Practical Strongly Invisible and Strongly Accountable Sanitizable Signatures**

[32] M. T. Beck, J. Camenisch, D. Derler, S. Krenn, H. C. Pöhls, K. Samelin, and D. Slamanig. Practical Strongly Invisible and Strongly Accountable Sanitizable Signatures. In *Proc. of Australasian Conference on Information Security and Privacy (ACISP 2017)*, volume 10342 of *LNCS*, pages 437–452. Springer, 2017. URL `https://doi.org/10.1007/978-3-319-60055-0_23` [last accessed: Sep. 2017]

nº 58 **Towards compactly encoded signed IoT messages**

[385] H. C. Pöhls and B. Petschkuhn. Towards compactly encoded signed IoT messages. In *Proc. of IEEE International Workshop on Computer-Aided Modeling Analysis and Design of Communication Links and Networks (IEEE CAMAD 2017)*, pages 1–6. IEEE, 2017. URL `https://doi.org/10.1109/CAMAD.2017.8031622` [last accessed: Sep. 2017]. URL `http://henrich.poehls.com/papers/2017_PoehlsPetschkuhn_IoT_signature_encoding_CAMAD.pdf` [last accessed: Sep. 2017]

nº 59 **Position Paper: The Past, Present, and Future of Sanitizable and Redactable Signatures**

[49] A. Bilzhause, H. C. Pöhls, and K. Samelin. Position Paper: The Past, Present, and Future of Sanitizable and Redactable Signatures. In *Proc. of International Conference on Availability, Reliability and Security (ARES 2017)*, pages 87:1–87:9. ACM, 2017. URL `http://henrich.poehls.com/papers/2017_BilzhausePoehlsSamelin_ARES17_RSS_SSS_PastPresentFuture.pdf` [last accessed: Sep. 2017]. URL `http://dx.doi.org/10.1145/3098954.3104058` [last accessed: Sep. 2017]

B — Glossary of Terms, Translations and Acronyms

Acronym	English term	German term
Directive 95/46/EC	Directive 95/46/EC on the protection of individuals with regard to the processing of personal data and on the free movement of such data [174]	
Directive 1999/93/EC	Directive 1999/93/EC on a Community framework for electronic signatures [175]	
Regulation 45/2001	Regulation 45/2001 on the protection of individuals with regard to the processing of personal data by the Community institutions and bodies and on the free movement of such data [176]	
Directive 2009/140/EC	Directive 2009/140/EC of 25 November 2009 amending Directives 2002/21/EC on a common regulatory framework for electronic communications networks and services, 2002/19/EC on access to, and interconnection of, electronic communications networks and associated facilities, and 2002/20/EC on the authorisation of electronic communications networks and services [180]	
Directive 2010/45/EU	Council Directive 2010/45/EU amending Directive 2006/112/EC on the common system of value added tax as regards the rules on invoicing [470]	
Regulation 910/2014 eIDAS	Regulation (EU) No 910/2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC [182]	
AC	anonymous credentials	
agent (legal)	from contract law: agent is the person who is authorized to act for another (denoted with the term 'principal') through employment, by contract or apparent authority; see for example § 164 BGB	Bevollmächtigter

Acronym	English term	German term
ANSI	American National Standards Institute	
AO	Fiscal Code of Germany (AO) [148]	Abgabenordnung
ArbGG	German Labour Court Act	Arbeitsgerichtsgesetz
	basic right	Grundrecht
BDSG	German Data Protection Act [91]	Bundesdatenschutzgesetz
BERR	UK's Department for Business Enterprise & Regulatory Reform (BERR)	
BFH	German Federal Fiscal Court	Bundesfinanzhof[1091]
BGB	German Civil Code [205]	Bürgerliches Gesetzbuch
BGH	German Federal High Court of Justice	Bundesgerichtshof
	blanket statement	Blankett, bzw. Blanketterklärung[1092]
BNetzA	German Federal Network Agency for Electricity, Gas, Telecommunications, Post and Railwal	Bundesnetzagentur[1093]
BPML	business process modelling language	
BSI	German Federal Office for Information Security	Bundesamt für Sicherheit in der Informationstechnik
BVerfG	German Supreme Constitutional Court	Bundesverfassungsgericht[1094]
BVerwG	German Federal Administrative Court	Bundesverwaltungsgericht[1095]
CA	certification authority	
CDA	HL7 Clinical Document Architecture [156]	
CDSS	classic digital signature scheme	
CFR	Code of Federal Regulations	
	civil proceedings	Zivilprozess[1094]
DSM	controllable local systems (smart grid)	
	doubts against evidence	Beweiseinreden[1096]
DLP	discrete logarithm problem	
DSM	demand side manager (smart grid)	
DSO	distribution system owner (smart grid)	
DSS	digital signature scheme (in the cryptographic sense)	
ECC	elliptic curve cryptography	
ECHR / EGMR	European Court of Human Rights (ECHR)	Europäischer Gerichtshof für Menschenrechte (EGMR)

[1091] Translation from `http://www.bundesfinanzhof.de/content/information-english` [last accessed Jan. 2016].

[1092] Own 'descriptive' translation as the otherwise used sources [70, 152] where suggesting no direct translations. However, 'blank signature' was found to translate to 'Blankett' in [225].

[1093] Translation from `http://www.bundesnetzagentur.de/cln_1412/EN/General/Bundesnetzagentur/Bundesnetzagentur-node.html;jsessionid=C1A67C10E6085047CF83B4199A0CDF97` [last accessed Jan. 2016].

[1094] German-English translation according to Langenscheidt Recht Englisch [70].

[1095] Translation from `http://www.bverwg.de/informationen/english/federal_administrative_court.php` [last accessed Jul. 2016].

[1096] Own 'descriptive' translation of the term used by *Grigorjew* [219] as other potential direct translations from [70, 152] if given offered a different connotation; see the discussion in footnote 973 on page 480.

Acronym	English term	German term
ECJ / EuGH	European Court of Justice (ECJ)	Europäischer Gerichtshof (EuGH)
EGVP	electronic court and administration postbox	Elektronisches Gerichts- und Verwaltungspostfach[1097]
eIDASDG		Gesetz zur Durchführung der Verordnung (EU) Nr. 910/2014 des Europäischen Parlaments und des Rates vom 23. Juli 2014 über elektronische Identifizierung und Vertrauensdienste für elektronische Transaktionen im Binnenmarkt und zur Aufhebung der Richtlinie 1999/93/EG (eIDAS-Durchführungsgesetz) [98]
ENISA	European Union Agency for Network and Information Security	
ESI	electronically stored information	
ETSI	European Telecommunications Standards Institute	
EU	European Union	
EU-CMA	existential forgery under chosen message attack	
	expert opinion	Sachverständigengutachten[1098]
FGO	German Code of Procedure for Fiscal Courts	Finanzgerichtsordnung
FOIA	Freedom of Information Act of UK [367] or US [490]	
	food business operator	Lebensmittelunternehmer [Art. 3 of Regulation (EC) No 178/2002]
EU General Data Protection Regulation GDPR	Regulation (EU) 2016/679 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) [184]	
GG	German Basic Law	Grundgesetz[1098]
GoBD	principles regarding the proper keeping and preservation of books, records and documents in electronic format and regarding data access	Grundsätze zur ordnungsmäßigen Führung und Aufbewahrung von Büchern, Aufzeichnungen und Unterlagen in elektronischer Form sowie zum Datenzugriff [83][1099]
GUI	graphical user interface	
HAN	home area network (smart grid)	
HCI	human computer interaction	
HGB	German Commercial Code	Handelsgesetzbuch[1098]

[1097] Own translation into English.

[1098] German-English translation according to Langenscheidt Recht Englisch [70].

[1099] Translation from http://www.sonntag-partner.de/en/news/go/gobd-new-bmf-letter-on-it-supported-accounting-systems/ [last accessed Jan. 2016].

Acronym	English term	German term
HHS	US Department of Health & Human Services	
	Hierarchy of Norms	Normenhierarchie[1100]
HMD	health monitoring device	
ICO	UK Information Commissioner's Office	
ICT	information and communication technology	
IEC	International Electrotechnical Commission	
IFG	German Federal Act Governing Access to Information held by the Federal Government (Freedom of Information Act)	Gesetz zur Regelung des Zugangs zu Informationen des Bundes (Informationsfreiheitsgesetz) [92]
INT	interactive non-public accountability as defined in Sec. 6.4.1 on page 150	
IP	intellectual property	
IPR	intellectual property right	
ISO	International Organization for Standardization	
IT	information technology	
ITU	International Telecommunication Union	
	jurisprudence	Rechtswissenschaft[1101]
	legal certainty	Rechtssicherheit[1100]
	legal effect	Rechtswirkung[1100]
	legal practice	Rechtssprechung[1102], Rechtspraxis[1100]
	legal text	Rechtstext / Gesetzestext[1101]
LMN	local metrological network (smart grid)	
MHT	Merkle hash tree (see [338])	
MSS	malleable signature scheme	Editierbare Signaturen
NIST	National Institute of Standards and Technology (US)	
OECD	Organisation for Economic Co-operation and Development	
OLG	German higher regional court	Oberlandesgericht[1100]
PET	privacy-enhancing technology	Privatheit fördernde Technik [371]
PGW	privacy gateway	

[1100] German-English translation according to Langenscheidt Recht Englisch [70].
[1101] Translation from *Dietl/Lorenz* - Woerterbuch Recht Wirtschaft und Politik [152].
[1102] Translation taken from `http://dict.leo.org` [last accessed Jan. 2016].

Acronym	English term	German term
PII	personally identifying information Note, PII is used in this thesis as a synonym to 'personal data'.	
PIN	personal identification number	
PKCS	Public Key Cryptography Standard [422]	
PKI	public key infrastructure	
POD	proof of delivery	Empfangsbekenntnis [§ 368 BGB]
PPT	probabilistic polynomial time	
	legal presumption of authenticity	gesetzliche Vermutung der Echtheit[1103]
	prevailing opinions of legal theorists	Literaturmeinung[1104] (cmp. Rechtssprechung, Rechtspraxis)
	prima facie	Anscheinsbeweis der Echtheit[1105]
	probative value	Beweiswert[1106]
	prosecution	Staatsanwaltschaft[1107]
PSS	probabilistic signature scheme	
PUB	non-interactive public accountability as proposed in Sec. 6.4.3 on page 153	
RAM	random access memory	
	relationship, external	Aussenverhältnis[1107]
	relationship, internal	Innenverhältnis[1107]
ReSCUeIT	German/French research project for IT supported secure supply chain managment	Deutsch-Fränzösisches Forschungsprojekt mit dem dem Titel '*R*obustes und v*e*rfügbares *SC*M - *U*nt*e*rstüzende *IT*-Plattform'
ROM	random oracle model	
RSA	Rivest–Shamir–Adleman algorithm	
RSS	redactable signature scheme	
SG	smart grid	
SGB	German Social Insurance Code	Sozialgesetzbuch
SHA	Secure Hash Algorithm	
SigG	German Signature Law	Signaturgesetz [93]
SigVO	German Signature Regulation	Signaturverordnung [94]
SM	smart meter (smart grid)	
SMGW	smart metering gateway (smart grid)	
SMO	smart meter operator (smart grid)	

[1103] Translation following the translations of the term 'presumption of ...' from *Dietl and Lorenz* - Woerterbuch Recht Wirtschaft und Politik [152].

[1104] Translation taken from `http://de.pons.com/open_dict/entries/Edeen32136653-8` [last accessed Jan. 2016].

[1105] Translation from *Dietl/Lorenz* - Woerterbuch Recht Wirtschaft und Politik [152].

[1106] Translation according to Langenscheidt Recht Englisch [70] and *Dietl and Lorenz* [152].

[1107] German-English translation according to Langenscheidt Recht Englisch [70].

Acronym	English term	German term
$\mathcal{SS}$	signature scheme, like in $contingent\mathcal{SS}$	
SSCD	secure signature creation device [Directive 1999/93/EC]	
QSCD	qualified electronic signature creation device [Regulation 910/2014]	
SSS	sanitizable signature scheme	
	statutory evidentiary presumption	Gesetzliche Beweisvermutung[1108]
StGB	German Criminal Code [89]	Strafgesetzbuch
StPO	German Code of Criminal Procedure	Strafprozessordnung
TLSs	Trusted Lists of Certification Service Providers [170]	
TRIPS	Trade-Related Aspects of Intellectual Property Rights	
TSP	trust service provider (TSP)	Zertifizierungsdiensteanbieter (ZDA)
TTP	trusted third party	
UC	universal composability	
US	United States of America	
U.S.C.	The United States Code[1109]	
UStG	German Value Added Tax Act [90, outdated version from 2005]	Umsatzsteuergesetz
VAT	value added tax	
VDG		Vertrauensdienstegesetz [151]
VGH	German superior (state) administrative court	Oberverwaltungsgericht[1110]
VIG		Gesetz zur Verbesserung der gesundheitsbezogenen Verbraucherinformation (Verbraucherinformationsgesetz - VIG)
VwGO	German Code of Administrative Court Procedure	Verwaltungsgerichtsordnung
VwVfG	German Administrative Procedure Act	Verwaltungsverfahrensgesetz
WHO	World Health Organization	Weltgesundheitsorganisation
WTO	World Trade Organization	Welthandelsorganisation
WYSIWYS	‘what you see is what you sign’	
XML	extensible markup language	
ZDA	trust service provider (TSP)	Zertifizierungsdiensteanbieter (ZDA)
ZPO	German Code of Civil Procedure (latest version [99], older version [96])	Zivilprozessordnung (n.F. [99], a.F. [96])

[1108] Translation from [398].

[1109] Translation from `http://uscode.house.gov/browse.xhtml` [last accessed: Sept. 2017].

[1110] German-English translation according to Langenscheidt Recht Englisch [70].

C List of Figures

1. Snippet from a supply chain document containing the original information 2
2. Snippet where information got cloaked/blackened/removed/ redacted/sanitized 2
3. Original supply chain document: waybill issued by freight sender 8
4. Subsequently modified document: recipients acknowledge receipt on the waybill creating the proof of delivery 8
5. Document is split into several defined blocks; sender's signature authorizes only specific blocks to be subsequently modified by only the forwarder and the recipient 9
6. Subsequently modified document contains verifiable signatures of sender, forwarder and receiver as only authorized modifications occurred 10
7. Unauthorized modifications, even when done by authorized parties, invalidate the signature(s) 10
8. Excerpt from a food contamination report issued by an authorized testing laboratory; . 11
9. Screenshot showing Adobe Acrobat Pro Version 9 on Mac OSX with the output when validating a signed PDF with a credit note from Amazon EU S.á r.l. 13
10. European Hierarchy of Norms and the legal texts discussed in Chapter 4 and in Chapter 8 15
11. The six steps of the methodology and the output of this thesis 19
12. Mapping cryptographic definition of a digital signature to legal terminology from Regulation 910/2014 (eIDAS) 30
13. (a) In-line, (b) on-line, and (c) off-line third parties by [337]; dotted lines denote communications carried out prior to protocol run; same information can also be found in ISO 14516 [244, p. 4–5] 32
14. Trust (dashed line) and verification key distribution in the classic signature process; Initially the signer's verification key is certified once involving the trusted third party . 33
15. Workflow between the involved parties in the classic signature process; Certificate Issuer as a TTP is off-line and not involved in signature generation and verification 33
16. Results after different types of modification 46
17. Workflow to sanitize "SECRET" into "SECure"; Verification needs just Signer's public verification key; Certificates can be **previously** issued by a TTP; TTP is offline and not involved in sanitization 54
18. Workflow to redact "SEC"; Verification needs just Signer's public verification key; Certificate can be **previously** issued by a TTP; TTP is offline and not involved in redaction 54
19. The four involved roles in a malleable signature and their trust relations 56
20. Differences of CDSS, existing secure SSS and RSS in their ability to select authorized actions and authorized principals. 57
21. High-level view of the privacy game (following *Brzuska et al.* [64]): The adversary should not be better than guessing when deciding which of the adversary provided input message produced the black-box's output. 60
22. Transparent vs. visible redactions 61
23. Different states that a Verifier can differentiate between in (a) transparent or (b) publicly accountable malleable signature scheme 62
24. Example of transparent modifications to calendar entries: (a) original, (b) transparently sanitized descriptions, (c) transparently redacted complete entries 63
26. Overview of legal texts analysed in Chapter 4; dashed arrows show influences 70
27. Example for a redacted document of official authority from an answer to a US FOIA request; taken from [161] 76
28. Existential unforgeability under chosen message attack (EUF-CMA) Experiment . . . 126

29. Left: Original; Middle: `Only_upper_half_of_1st_page_copied`, title and authors still preserved but lost total number of pages; Right: `Only_1st_page_-pagenumbers_adjusted`, title and authors still preserved but lost total number of pages 132
30. Venn-style diagram of dividing the set of all modifications into subsets of 'unmodified', 'modified in an unauthorized way' and 'modified in an authorized way'; modifications and original from the example are indicated by a • as elements of those sets. Note, Fig. 31 shows the final tri-section. 133
31. Venn-style diagram of differentiating documents into disjunct sets of 'unmodified' and 'modified'; Further dissecting the 'modified' subset into 'modified in an unauthorized way' and 'modified in an authorized way' subsets 135
32. Comparison of selected integrity definitions along two aspects 142
33. Lattice of the three levels of increasing strictness for tolerable subsequent modifications 143
34. Malleable signature schemes (MSS) and classical digital signature schemes (CDSS) regarding the first aspect of the extended integrity definition: Three levels (UCA, ACA, and NCA) of increasing strictness for tolerable subsequent modifications 144
35. Integrity details: Several levels in two groups to differentiate the increasing level of details in the detection; (a) detection of occurred subsequent modifications (*CD); (b) detection of potential future subsequent modifications (*FD) 147
36. Four quadrants; on the x-axis the levels of Detectability and thus Accountability; on the y-axis message-level or block-level in general; to the outside an increased level of detail from the complete message to blocks 160
37. Classical digital signature schemes (CDSS) based on a secure cryptographic hash-function, RSASSA-PSS from PKCS-v2.2 [422], offer NCA–1CD–PUB integrity, because Verifiers detect any subsequent modification without distinguishing between authorized or unauthorized modification and verification proves the Signer's accountability for unmodified messages non-interactively, i.e. by using just the Signer's public verification key 162
40. Overview of legal texts analysed in Chapter 8; dashed arrows show influences 192
42. Figure from *Brzuska et al.* [66] showing that the adversary can adaptively and repeatedly query all oracles 219
43. Privacy Experiment for RSS on tree structured data from *Brzuska et al.* [66]; the algorithm names have been slightly changed and an index j was removed to simplify the presentation compared to [66] 220
45. Indistinguishability (IND-CCA2) Experiment following [42] 226
47. Unforgeability Experiment for SSS based on *Brzuska et al.* [64] and *Gong et al.* [216] . 261
48. Immutability Experiment for SSS; based on *Brzuska et al.* [64] and *Gong et al.* [216] . 262
49. Immutability Experiment for SSS as originally stated by *Brzuska et al.* [64] 263
50. Immutability Experiment for SSS revised 263
51. Standard Privacy Experiment for SSS based on *Brzuska et al.* [64] including additions from *Krenn et al.* [294] 265
52. Proof-Restricted Transparency Experiment for SSS following *Brzuska et al.* [67] . . . 267
53. Proof-Restricted Transparency Experiment for SSS based on *Brzuska et al.* [67] with additions from *Krenn et al.* [294] and *Camenisch et al.* [107] 268
54. Unlinkability Experiment for SSS based on *Brzuska et al.* [67] 270
55. Interactive Non-Public Signer-Accountability Experiment for SSS based on *Brzuska et al.* [64] and *Gong et al.* [216] 271
56. Interactive Non-Public Sanitizer-accountability Experiment for SSS based on *Brzuska et al.* [64] and *Gong et al.* [216] 272
57. Unforgeability Experiment for RSS 287
58. Unforgeability Experiment for RSS using a set-like notation 287
59. Immutability Experiment for RSS 288
60. Standard Privacy Experiment for RSS based on *Brzuska et al.* [66] including additions from *Krenn et al.* [294] 289
61. Transparency Experiment for RSS 290
62. Sample tree with three nodes 297
63. Original tree T with traversal numbers 299
64. Transitive closure of T from Fig. 63, i.e. $\mathrm{span}_{\vdash}(T)$ 299

65. Redacting T from Fig. 63 into T' 299
66. Mix and match attack 299
67. Original list of student's name and grade, ordered by grade 306
68. Redacting all list elements containing the grade 306
69. Not redacted list elements remain ordered by grade 306
70. Goal: Remove all list elements containing grades and order among elements 306
71. The operation ‘merge’ is an explicit inverse of the redact operation; merging allows to generate validly signed variants from any two valid, but modified variants if and only if they have a common ancestor (original or redacted variant) 307
72. Message-level Non-Interactive Public Accountability (PUB) Experiment for SSS 311
73. Strong Message-level Non-Interactive Public Accountability (PUB) Experiment for SSS following *Krenn et al.* [294] 312
74. Block-level Non-Interactive Public Accountability (PUB) Experiment for SSS 313
75. Transparency property in contrast to the new property of non-interactive public accountability 314
76. Visual representation of schemes that support forms of *non-interactive public* accountability of both the Signer and the Sanitizer on *message-level* 317
77. Visual representation of schemes that support forms of *public non-interactive* accountability of both the Signer and the Sanitizer on *block-level*, which *includes message-level* properties 318
78. Group-level Non-Interactive Public (PUB) Accountability Experiment for SSS 322
79. Group-level Interactive Non-Public Signer-Accountability (INT) Experiment for SSS 323
80. Group-level Interactive Non-Public Sanitizer-Accountability (INT) Experiment for SSS 323
81. Visual representation of the schemes that support block-level interactive non-public (INT / transparent) accountability 325
82. Strengthened Unlinkability Experiment for SSS 327
83. Standard Unlinkability Experiment for SSS by *Brzuska et al.* [67] 327
84. Standard Privacy Experiment for SSS adjusted for block-level non-interactive public accountability 330
85. Construction idea: Blocks 2,6,7 of m are fixed and together with $\mathsf{pk}_{\mathsf{san}}$ and ADM signed by signer (σ_{fix}); the complete message m is signed by either the signer or the sanitizer (σ_{full}) 333
86. Construction 2: Block-level public accountability due to the block-level signatures 334
87. $pubacc\mathcal{SSS}$ offers ACA – 1CD – PUB integrity; $blockacc\mathcal{SSS}$ only offers ACA – ≥1CD,< BCD – PUB integrity 338
88. $pubacc\mathcal{SSS}^{Block}$ offers ACA – BCD – PUB integrity 339
89. Collision-Resistance against Random Tagging Attacks Experiment [64] 342
90. $blockacc\mathcal{SSS}^{PUB}$ offers ACA – ≥1CD,< BCD – PUB with added information about groups of blocks 350
91. $blockacc\mathcal{SSS}^{INT}$ offers ACA – ≥1CD,< BCD – INT 351
92. $unlinkable\mathcal{SSS}$ offers ACA – 1CD – PUB with added perfect unlinkability 358
93. $pubacc\mathcal{SSS}^{Block}$: Interactions between host and secure smart card during the Sign algorithm; from [370] but notation adjusted 360
94. $pubacc\mathcal{SSS}^{Block}$: Interactions between host and secure smart card during the Sanit algorithm; from [370] but notation adjusted 360
95. Signer-Accountability Experiment for ARSS 367
96. Sanitizer-Accountability Experiment for ARSS 368
97. Public Accountability Experiment for ARSS 368
98. Visual representation of an RSS that supports *non-interactive public* accountability of both the Signer and the Sanitizer on *message-level* (ARSS) 369
99. Unforgeability Experiment for MRSS 371
100. Privacy Experiment for MRSS 371
101. $merge\mathcal{RSS}$ Transparency Experiment 372
102. Merge Privacy Experiment for MRSS 372
103. Merge Transparency Experiment for MRSS 373
104. Update Privacy Experiment for MRSS 373
105. Update Transparency Experiment for MRSS 374
106. Unlinkability Experiment for RSS based on *Brzuska et al.* [66] 376

107. $priv\mathcal{RSS}$ Privacy Experiment 378
108. $priv\mathcal{RSS}$ Transparency Experiment 379
109. $priv\mathcal{RSS}$'s construction is built upon two ordered lists of pairwise unique random numbers signed by an order protecting transparent RSS 382
110. $priv\mathcal{RSS}$ offers ACA – UCD integrity, because Verifiers can only detect unauthorized subsequent modifications and there is no accountability in $priv\mathcal{RSS}$; can be added by ARSS 387
111. Original, Expanded and valid Redactions of an unordered tree with an allowed non-leaf redaction 389
112. Indistinguishable Experiment for Accumulators 392
113. $flex\mathcal{RSS}$ offers ACA – UCD integrity 398
114. Disclosure Secure Experiment for RSS that offer Consecutive Redaction Control like $struct\mathcal{RSS}$ 400
115. $struct\mathcal{RSS}$ offers ACA – UCD integrity, because Verifiers can only detect unauthorized subsequent modifications and there is no accountability in $struct\mathcal{RSS}$; can be added by ARSS 409
116. Strong Collision-Resistance 411
117. $merge\mathcal{RSS}$ offers ACA – UCD – INT integrity, because Verifiers can detect always only unauthorized subsequent modifications as there is no differentiation between an original and a message modified with authorization inside the transparent $merge\mathcal{RSS}$ that would allow to hold a Sanitizer accountable 416
118. $pubacc\mathcal{RSS}$ offers ACA – 1CD – PUB integrity if based on an SSS that offers ACA – 1CD – PUB, because Verifiers are able to detect authorized subsequent redactions for which the redactor is accountable through its dedicated key pair 423
119. Collision-Resistance Experiment for Accumulator's Public Key Generator within a Semi-Trusted Setup 426
120. Collision-Resistance Experiment for Accumulator's Parameter Generator within a Semi-Trusted Setup 426
121. $merge\mathcal{RSS}$: Interactions between host and smart card during KeyGen algorithm . . . 429
122. $merge\mathcal{RSS}$: Interactions between host and smart card during the Sign algorithm . . . 429
123. Weak Immutability Experiment for SSS 433
124. Strengthened Privacy Experiment for SSSs 434
125. Weak Privacy Experiment for RSS 435
126. Strong Privacy Experiment for RSS 435
127. Weak Block-level Non-Interactive Public Accountability Experiment for SSS 436
128. Overview and hierarchy of legal norms; Left: those used to identify the probative value of malleable signatures; Right: those used to assess the privacy requirements 456
130. Legally required integrity protection for mechanisms in which authorized subsequent changes are allowed (ACA integrity) in fulfilment of Requirement 3: If one or more subsequent modification (authorized or unauthorized) has occurred, this must be detected as modified. 465
131. Legally required integrity protection according to the mandatory parts of Requirement 3 and Requirement 4 for mechanisms in which authorized subsequent changes are allowed (ACA integrity) 467
132. Cases of CDSS signature verification results 472
133. Cases of MSS properties and signature verification results 473
134. Screenshot of SAP's modelling GUI [352] 491
135. ReSCUeIT's example food supply chain with four partners (from left/bottom to right/top: receiver, logistics, producer and food laboratory); 492
136. Runtime in milliseconds (ms) of the different operations carried out on a waybill file by the web services implemented in ReSCUeIT 501
137. Example workflow for e-health use-case: integrity and origin of remaining data remains verifiable and protected against active attacks; redaction needs no keys 502
138. $merge\mathcal{RSS}$ Performance: runtime in seconds of signing a 100-element message with $4,096$ and $2,048$ bit keys on an Intel Core i7-4790 CPU 505
139. Trust towards components by SG stakeholders and privacy-aware household 507

140. System structure with the privacy gateway (PGW) trusted by the residential customer (RC) in front of smart metering gateway (SMGW) trusted by the smart metering operator (SMO) 510
141. Simplified overview of supply participants in the production of egg noodles 513
142. Sticky signatures: redacted signed data and credentials of inputs are included 513
143. Flow of a request: The data subject directly sends a `ROAR` requests to `C4`, which results in further `ROAD` requests 516
144. `ROAR` response message for a `ROAR` request to `C4` of the case shown in Fig. 143 517
145. Example of splitting a redactable signed database table and keeping verifiable integrity 519
146. Bank cheque with the amount made out in blank: Example of a form to be filled-in later as some fields have no values written in it (circled just for highlighting); as the entire cheque is already signed it is assumed to create a blanket statement 521
148. Example of a paper based form with fields that are not supposed to be filled in when the first signature is made on the form; excerpt from [454] 526
149. Contingency Experiment 529

D — List of Tables

1. Integrity related computer science definition: Treatment of authorized modifications and requirement to detect any subsequent modification 137
2. Origin authentication and non-repudiation related computer science definition and their treatment of authorized modifications and requirement to make Sanitizer accountable for any subsequent modification 139
3. Existing integrity-related computer science definitions restated with the new aspects . . 159
4. Relation between legal analysis results and technical requirements elicited in this thesis for an increased probative value; *rec.=recommended*; *opt.=optional* 185
5. Relation between technical analysis results and technical requirements elicited in this thesis for an increased probative value; *rec.=recommended*; *opt.=optional* 186
6. Matrix indicating a clear link between a signature functionality from the official interpretation of the German signature legislation from [149] and [147] and the technical requirements elicited in this thesis for an increased probative value; *rec.=recommended*; *opt.=optional* 187
7. Overview of the analysed legal texts and their discussion of the required confidentiality protection as well as their use or definition of terminology related to sanitization or redaction; "–" means no or not discussed; "(✓)" means indirectly 209
8. Mapping the titles of the 11 privacy principles from ISO 29100 and those from OECD discussed in Sec. 8.2.6 228
9. Selected existing schemes which have shortcomings by not achieving standard privacy (as defined in Definition 156 and in Definition 174) or have shortcomings with privacy related properties; for an explanation of the attacks please see the respective stated papers; for new attacks or more explanation see indicated subsections. 295
10. Selected existing schemes where shortcomings in integrity protection or related properties are presented 298
11. Overview of new SSS properties and the three new SSS schemes presented in this thesis 309
12. $pubacc\mathcal{SSS}$: Performance of message-level scheme from Construction 1; ℓ is the number of blocks; median runtime in ms 336
13. $pubacc\mathcal{SSS}^{Block}$: Performance of block-level scheme from Construction 2; ℓ is the number of blocks; median runtime in ms 336
14. $blockacc\mathcal{SSS}^{INT}$: Performance of Construction 3; ℓ is the number of blocks; λ is the security parameter in Bit; median runtime in ms 346
15. $blockacc\mathcal{SSS}^{PUB}$: Performance of Construction 4; ℓ is the number of blocks; λ is the security parameter in Bit; median runtime in ms 347
16. $unlinkable\mathcal{SSS}$: Performance of Construction 5; ℓ is the number of blocks; median runtime in ms 355
17. Performance of SSS prototypes on secure smart card (“SmartC@fé® Expert 4.x” [206]); median runtime in seconds 361
18. Overview of new RSS properties and the five new RSS schemes presented in this thesis 363
19. *Brzuska* et al.: Low Branching Factor n; Median Runtime in ms 385
20. *Kundu and Bertino*: Low Branching Factor n; Median Runtime in ms 385
21. *Brzuska* et al.: High Branching Factor n; Median Runtime in ms 385
22. *Kundu and Bertino*: High branching factor n; Median Runtime in ms 385
23. $flex\mathcal{RSS}$ Performance; median runtime in ms 395
24. $struct\mathcal{RSS}$ instantiation’s median runtime in ms 406
25. Performance of $merge\mathcal{RSS}$ and $pubacc\mathcal{SSS}$ (for comparison) prototype on smart card; median runtime in seconds 430

26. Overview of proposed SSS and RSS properties required to separate secure RSS from secure SSS 431
27. All schemes proposed in this thesis are practically usable for protecting private data, because they are — at least — private as defined by *Brzuska et al.* [64]; this fulfils Requirement 9 460
28. Mandatory requirements 1-7 for increased probative value and their fulfilment by proposed schemes; namely $pubacc\mathcal{SSS}^{Block}$ and $pubacc\mathcal{RSS}$ achieve all mandatory and the recommended, but optional, requirements 462
29. All proposed schemes (and all their constructions) fulfil Requirement 1 463
30. All proposed constructions fulfil the mandatory parts of Requirement 2 with different protection scopes and different flexibility to define ADM; Only constructions with ≥1CD fulfil Requirement 3 464
31. All schemes proposed in this thesis are electronic signatures following Regulation 910/2014470
32. Schemes achieving to have a verification outcome in Case 1 are qualified electronic signatures following Regulation 910/2014 479
33. Total speed in milliseconds (ms) of the different operations carried out on a waybill file used in ReSCUeIT 501
34. $merge\mathcal{RSS}$ Performance: runtime in seconds of the KeyGen, Redact, Merge operations 505
35. $merge\mathcal{RSS}$ Performance: average runtime in seconds of signing 100- and 1,000-element messages 506
36. $merge\mathcal{RSS}$ Performance: average runtime in seconds of updating 100 and 1,000 elements to a message 506
37. $merge\mathcal{RSS}$ Performance: average runtime in seconds of verifying 100- and 1,000-element messages 506
38. Ranking of proposed constructions according to their direct fulfilment of all recommended and mandatory requirements 538