Certification and Efficient Proofs of Committed Topology Graphs

Thomas Groß
School of Computing Science, Newcastle University, UK
thomas.gross@ncl.ac.uk

***Abstract*—Digital signature schemes are a foundational cryptographic building block in certification and the projection of trust. Based on a signature scheme on committed graphs, we propose a toolkit of certification and proof methods to sign committed topology graphs and to prove properties of their certificates in zero-knowledge. This toolkit allows an issuer, such as an auditor, to sign the topology representation of an infrastructure. The prover, such as an infrastructure provider, can then convince a verifier of topology properties, such as partitions, connectivity or isolation, without disclosing the structure of the topology itself. By that, we can achieve the certification of the structure of critical systems, such as infrastructure clouds or outsourced systems, while still maintaining confidentiality. We offer zero-knowledge proofs of knowledge for a general specification language of security goals for virtualized infrastructures, such that high-level security goals can be proven over the topology certificate. Our method builds upon the Camenisch-Lysyanskaya signature scheme, is based on honest-verifier proofs and the strong RSA assumption.**

I. Introduction

Digital signature schemes are foundational cryptographic primitives, in particular to ensure the primary security property of integrity. From their conception [1], digital signature schemes have been employed to sign messages or committed, hidden messages [2], as well as to establish the integrity of systems and their components via certification of software or Direct Anonymous Attestation (DAA) [3]. Digital signatures being used to establish the attestation of a system is particularly relevant when a tenant delegates storage, networking or computation to a provider, such as in outsourcing or cloud computing. In this work, we focus on the question how digital signatures can ensure integrity in a delegated computation scenario.

The tenants will question the integrity of the systems in which their resources are hosted. The systems are typically large topologies with flat hierarchies, by which structural properties, interconnectivity and isolation are important for the security of the tenants' sub-systems and the system at large. The tenants will naturally expect the provider to convince them that the system is well-structured and that their own resources are properly isolated from other tenants. However, this fundamental integrity requirement of the tenants is at odds with the confidentiality requirement of the provider.

The provider of such a system requires the confidentiality of the blue-print of the system at large. It aims to protect the tenants from exposure and to ensure that the tenants own confidentiality requirements on their sub-systems are fulfilled. Therefore, we ask: How can a provider convince a verifier that the topology fulfills security properties, such as tenant isolation, without disclosing other information about the topology itself?

In this work, we pursue the research hypothesis that a signature scheme on committed graphs and efficient zero-knowledge proofs of knowledge thereon can solve this problem. They provide us with proofs of knowledge that show elaborate statements on topology security properties, while keeping the topology itself confidential. While past work in the integrity of systems focused on attestation of individual hosts and virtualized attestation of virtual machines, based on the Trusted Platform Module, we complement that work with the confidential attestation of the system structure. We convince a

verifier that the system is structured securely, while keeping the blueprint of the system secret.

The main contribution of this work is a framework of efficient zero-knowledge proofs of knowledge over signatures on committed graphs. Those proofs of knowledge directly apply to interesting security goals raised by tenants, as for instance expressed in the goal language *VALID*. The fundamental encoding and signature for undirected graphs is already proposed in a companion paper, which includes the proof that the signature scheme and corresponding proofs can express statements from NP-languages in a known-graph setting. In this work, we extend the encoding to directed graphs.

We establish general efficient proofs of knowledge for certification of topologies. Those proofs of knowledge allow to make statements over vertex sets, connectivity and isolation. Combined with known discrete-logarithm based proofs of knowledge [4], [5], [6], [7], [8], [9], [10], we obtain an expressive toolkit.

We believe that the topology graph signatures we present in this paper are a crucial building block to close the gap between existing attestation of individual components of an infrastructure (e.g., with DAA) and statements on the security of the entire infrastructure. In particular, we propose an efficient general method to prove that isolation goals of tenants are fulfilled, while keeping the topology itself confidential. This allows us to overcome the fundamental gap between the integrity requirements of the tenant and the confidentiality requirements of the provider.

A. Outline

In Section II, we discuss the preliminaries of our graph proof construction: Camenisch-Lysyaskaya signatures and Camenisch-Groß encoding. Based on the Camenisch-Groß encoding, we establish a canonical encoding for vertex- and edge-labeled graphs in Section IV. Having established the preliminaries, Section III introduce the interfaces for the graph signature scheme algorithms and the library of proof predicates we construct subsequently. The following sections focus on the implementation of these interfaces, where Section IV establishes the underlying encoding for undirected and directed graphs. Section V-B offers the core building blocks of proofs of representations as well as key generation and issuing of signatures on joint graphs. Section VI offers constructions for a library of graph proofs, starting from statements over vertex set, adjacency to connectivity and isolation statements. Section VII establishes the efficiency of the system. Section VIII compares this work with earlier proposals for transitive and homomorphic graph signatures and zero-knowledge proofs on graphs, while Section IX discusses future work.

II. Preliminaries

A. Assumptions

Special RSA Modulus A *special RSA modulus* has the form $N = pq$, where $p = 2p'+1$ and $q = 2q'+1$ are safe primes, the corresponding group is called *special RSA group*.

Strong RSA Assumption [1], [11], [6]: Given an RSA modulus N and a random element $g \in \mathbb{Z}_N^*$, it is hard to compute $h \in \mathbb{Z}_N^*$ and integer $e > 1$ such that $h^e \equiv g \bmod N$. The modulus N is of a special form pq, where $p = 2p' + 1$ and $q = 2q' + 1$ are safe primes. *Quadratic Residues* The set QR_N is the set of Quadratic Residues of a special RSA group with modulus N.

B. Integer Commitments

Damgård and Fujisaki [12] showed for the Pedersen commitment scheme [13] that if it operates in a special RSA group and the committer is not privy to the factorization of the modulus, then the commitment scheme can be used to commit to *integers* of arbitrary size. The commitment scheme is information-theoretically hiding and computationally binding.

The security parameter is ℓ. The public parameters are a group G with special RSA modulus N, and generators $(g_0, \ldots, g_m)$. In order to commit to the values $(V_1, \ldots, V_l) \in (\mathbb{Z}_n^*)^l$, pick a random $R \in \{0,1\}^\ell$ and set

$$C = \mathsf{Commit}(R, V_1, \ldots, V_l) = g_0^R \prod_{i=1}^{l} g_i^{v_i}.$$

C. Known Discrete-Logarithm-Based Proofs

In the common parameters model, we use several previously known results for proving statements about discrete logarithms, such as (1) proof of knowledge of a discrete logarithm modulo a prime [4] or a composite [12], [6], (2) proof of knowledge of equality of representation modulo two (possibly different) prime [7] or composite [8] moduli, (3) proof that a commitment opens to the product of two other committed values [14], [8], (4) proof that a committed value lies in a given integer interval [9], [8], [10], and also (5) proof of the disjunction or conjunction of any two of the previous [15]. These protocols modulo a composite are secure under the strong RSA assumption and modulo a prime under the discrete logarithm assumption.

Proofs as described above can be expressed in the notation introduced by Camenisch and Stadler [16]. For instance,

$$PK\{(\alpha, \beta, \delta) : y = g^{\alpha} h^{\beta} \wedge \tilde{y} = \tilde{g}^{\alpha} \tilde{h}^{\delta} \wedge (u \leq \alpha \leq v)\}$$

denotes a "*zero-knowledge Proof of Knowledge of integers α, β, and δ such that $y = g^{\alpha} h^{\beta}$ and $\tilde{y} = \tilde{g}^{\alpha} \tilde{h}^{\delta}$ holds, where $u \leq \alpha \leq v$,*" where $y, g, h, \tilde{y}, \tilde{g}$, and $\tilde{h}$ are elements of some groups $G = \langle g \rangle = \langle h \rangle$ and $\tilde{G} = \langle \tilde{g} \rangle = \langle \tilde{h} \rangle$. The convention is that Greek letters denote quantities of which knowledge is being proven, while all other values are known to the verifier. We apply the Fiat-Shamir heuristic [17] to turn such proofs of knowledge into signatures on some message m; denoted as, e.g., $SPK\{(\alpha) : y = g^{\alpha}\}(m)$. Given a protocol in this notation, it is straightforward to derive an actual protocol implementing the proof.

We introduce the following short-hands: i. A modulus statement $(\mathrm{mod}\ N)$ in the PK-header denotes the default modulus for subsequent non-range proof congruences. ii. An all quantifier $\forall i$ denotes that the secrets/terms in its scope are iterated over i. iii. We decompose proofs of knowledge statements in multiple steps and require referential integrity between the secrets of the steps. For example, statements $PK\{(\mu)\}$ and $PK\{(\nu)\}$ construct one compound proof of knowledge $PK\{(\mu, \nu)\}$.

D. Camenisch-Lysyanskaya Signatures

Let us introduce Camenisch-Lysyanskaya (CL) signatures in a Strong RSA setting [2].

Let $\ell_{\mathcal{M}}$, ℓ_e, ℓ_N, ℓ_r and L be system parameters; ℓ_r is a security parameter, $\ell_{\mathcal{M}}$ the message length, ℓ_e the length of the Strong RSA problem instance prime exponent, ℓ_N the size of the special RSA modulus.

Parameters. The scheme operates with a ℓ_N-bit special RSA modulus. Choose, uniformly at random, $R_0, \ldots, R_{L-1}, S, Z \in \mathrm{QR}_N$. The public key pk_{I} is $(N, R_0, \ldots, R_{L-1}, S, Z)$, the private key sk_{I} the factorization of the special RSA modulus.

Message space is the set $\{(m_0, \ldots, m_{L-1}) : m_i \in \pm\{0,1\}^{\ell_{\mathcal{M}}}\}$.

Signing algorithm. On input $m_0, \ldots, m_{L-1}$, choose a random prime number e of length $\ell_e > \ell_{\mathcal{M}} + 2$, and a random number v of length $\ell_v = \ell_N + \ell_{\mathcal{M}} + \ell_r$. Compute

$$A = \left(\frac{Z}{R_0^{m_0} \cdots R_{L-1}^{m_{L-1}} S^v} \right)^{1/e} \mod N.$$

The signature consists of (e, A, v).

The CL-signature scheme can be used to sign hidden committed attributes. A user U commits to values V in an integer commitment C and proves knowledge of the representation of the commitment. The issuer I verifies the structure of C and signs the commitment as defined above:

$$A = \left(\frac{Z}{C R_l^{m_l} \cdots R_{L-1}^{m_{L-1}} S^{v'}} \right)^{1/e} \mod N.$$

The user completes the signature as follows:

$$\sigma = (e, A, v) = (e, A, (v' + R))$$

Verification algorithm. To verify that the tuple (e, A, v) is a signature on message $(m_0, \ldots, m_{L-1})$, check that the following statements hold: $Z \equiv A^e R_0^{m_0} \cdots R_{L-1}^{m_{L-1}} S^v \pmod N$, $m_i \in \pm\{0,1\}^{\ell_{\mathcal{M}}}$, and $2^{\ell_e} > e > 2^{\ell_e - 1}$ holds.

Theorem 2.1: [2] The signature scheme is secure against adaptive chosen message attacks [18] under the strong RSA assumption.

Proving Knowledge of a Signature. A prover can prove that she possesses a CL-signature without revealing any other information about the signature (as well as use the primitives in Section II-C). The prover randomizes A: Given a signature (A, e, v), the tuple $(A' := AS^{-r} \bmod N, e, v' := v + er)$ is also a valid signature as well. Now, provided that $A \in \langle S \rangle$ and that r is chosen uniformly at random from $\{0,1\}^{\ell_N + \ell_\varnothing}$, the value A' is distributed statistically close to uniform over $\mathbb{Z}_N^*$. Thus, the user could compute a fresh A' each time, reveal it, and then run the protocol

$$PK\{(\varepsilon, \nu', \mu_0, \ldots, \mu_{L-1}) : \\ Z \equiv \pm R_0^{\mu_0} \cdots R_{L-1}^{\mu_{L-1}} A'^{\varepsilon} S^{\nu'} \pmod N \ \wedge \\ \mu_i \in \pm\{0,1\}^{\ell_{\mathcal{M}}} \ \wedge \ \varepsilon \in [2^{\ell_e - 1} + 1, 2^{\ell_e} - 1]\}$$

E. Set Membership from CL-Signatures

Set membership proofs can be constructed from CL-Signatures following a method proposed by Camenisch, Chaabouni and shelat [19]. For a set $\mathcal{S} = \{m_0, \ldots, m_i, \ldots, m_l\}$, the issuer signs all set members m_i in CL-Signatures $\sigma_i = (A, e, v)$ and publishes the set of message-signature pairs $\{(m_i, \sigma_i)\}$ integerly. To prove set membership of a value committed in C, the prover shows knowledge of the blinded signature σ_i' corresponding to the message m_i and equality of exponents with C. We explain this technique in detail in Appendix A and denote a set membership proof $\mu[C] \in \mathcal{S}$, which reads μ encoded in commitment C is member of set $\mathcal{S}$.

F. Camenisch-Groß Encoding

The Camenisch-Groß (CG) Encoding [20] gives the CL message space structure by encoding multiple binary and finite-set values into a single message, and we will use a similar paradigm to encode graphs efficiently. We explain the key principles briefly and give more details in Appendix B.

The core principle of the CG-Encoding is to represent binary and finite-set attribute values as prime numbers. It uses divisibility and coprimality to show whether an attribute value is present in or absent from a credential. The attribute values certified in a credential, say e_i, e_j, and e_l, are represented in a single message of the CL-Signature, by signing the product of their prime representative $E = e_i \cdot e_j \cdot e_l$ in an Integer attribute. The association between the value and the prime number of the encoding is certified by the credential issuer.

Divisibility/AND-Proof. To prove that a disclosed prime representative e_i is present in E, we prove that e_i divides the committed product E, we show that we know a secret μ' that completes the product:

$$PK\{(\mu', \rho) : \qquad D \equiv \pm(g^{e_i})^{\mu'} h^{\rho} \pmod N\}.$$

Coprimality/NOT-Proof. We show that one or multiple prime representatives are not present in a credential, we show coprimality. To prove that two values E and F are coprime, i.e., $\gcd(E, F) = 1$, we prove there exist integers a and b such that Bézout's Identity equals 1, where a and b for this equation do not exist, if $\gcd(E, F) > 1$.

$$PK\{(\mu, \rho, \alpha, \beta, \rho') : \qquad D \equiv \pm g^{\mu} h^{\rho} \pmod N \ \wedge \\ g \equiv \pm D^{\alpha} (g^F)^{\beta} h^{\rho'} \pmod N\}.$$

OR-Proof To show that a credential contains an attribute e that is contained in an OR-list, we show there exists an integer a such that $ae = \prod_i^{\ell} e_i$; if e is not in the list, then no such integer a as e does not divide the product. We use the notation $\alpha \subseteq \Xi$ for an OR-proof that α contains one or more values of Ξ.

III. Graph Signature Scheme and Proofs

We specify, for the first time, the abstract interface of a graph signature scheme and associated proofs over graph properties. The core signature scheme consists of four algorithms: Commit, Keygen, HiddenSign, and Verify.

$\mathsf{Commit}(\mathcal{G}; R)$ is a probabilistic polynomial-time algorithms, providing an Integer commitment commitment lifted to graphs. It takes as input a graph $\mathcal{G}$ encoded with the encoding $\mathsf{encode}(\mathcal{G})$ specified in Section IV and randomness R.

$\mathsf{Keygen}(1^{\ell}, params)$ is a probabilistic polynomial-time algorithm, which establishes the key setup for the graph signature scheme. It

takes as input the security parameter ℓ and the public parameters of the commitment scheme Commit. It outputs a key pair (pk, sk), where pk is the public key of the issuer and sk is the its secret key.

$\mathsf{HiddenSign}(C = \mathsf{Commit}(\mathcal{G}_\mathsf{U}; R), \mathcal{V}_\mathsf{U}, \mathcal{V}_\mathsf{I}, pk_\mathsf{I})$ is an interactive probabilistic polynomial-time algorithm between a user U and an issuer I. It is to sign a graph jointly contributed by a user sub-graph $\mathcal{G}_\mathsf{U}$ and an issuer sub-graph $\mathcal{G}_\mathsf{I}$. The common public input is a commitment on the user sub-graph $\mathsf{Commit}(\mathcal{G}_\mathsf{U}; R)$, disclosed connections points $\mathcal{V}_\mathsf{U}, \mathcal{V}_\mathsf{I}$, and the issuer's public key pk_I. The user's private input is his sub-graph $\mathcal{G}_\mathsf{U}$ and the commitment randomness R. The issuer's private input is his sub-graph $\mathcal{G}_\mathsf{I}$ and his secret key sk_I. The user's output is (σ, R'), that is, a signature of the joint graph $\sigma = \sigma(\mathcal{G}_\mathsf{U} \uplus \mathcal{G}_\mathsf{I})$ and a new randomness R' combining his commitment randomness R with the randomness of the issuer.

$\mathsf{Verify}(pk_\mathsf{I}, C, R', \sigma)$ is a polynomial-time signature verification algorithm. It takes as input the issuer's public key pk_I, the original commitment on the user's sub-graph C, the randomness R' and signature σ. It outputs 1 if the signature is verified, 0 otherwise.

In addition, we provide proof predicates for graph signatures (cf. with Table I): First, we provide an abstract predicate for the graph proof of representation graph and a proof of possession of the corresponding signature $\mathsf{possession}$, both implemented in Section V-B. Second, we consider sets of graph elements with set coverage cover, pair-wise disjointness $\mathsf{disjoint}$ and partition $\mathsf{partition}$. Third, we have predicates on connectivity edge and $\mathsf{connected}$ and its complement isolation $\mathsf{isolated} = \neg\mathsf{connected}$.We establish an implementation for zero-knowledge proofs of knowledge thereof in Section VI. The constructed proofs of knowledge can be combined with known discrete-logarithm based proofs of knowledge and composed to Boolean formulas with logical connectives $\wedge$ and $\vee$.

Remark 1 (Formal Cloud Security Assurance): The predicates predicates introduced in Table I correspond to the major predicates of the Virtualization Assurance Language for Isolation and Deployment (*VALID*) [21], a formal specification language of cloud security goals suitable for automated model checking [22]. *VALID* expresses goal states as a set (conjunction) of positive and negative facts constrained by a Boolean condition list. It uses terms, such as $\mathsf{edge}(\cdot,\cdot)$ or $\mathsf{connected}(\cdot,\cdot)$ to express alarm states on topology graphs. By establishing proof of knowledge predicates for this language, we allow a prover to convince a verifier that the topology is compliant with the verifier's *VALID*-specified security policy.

In the following, we will introduce the implementations for proofs of knowledge for these different functions successively. Section IV introduces the encoding of undirected and directed graphs itself. Section V introduces the key generation Keygen, the proof of representation graph and the issuing and verification algorithms $\mathsf{HiddenSign}$ and Verify. Subsequently, Section VI contains the constructions for the set and connectivity predicates.

IV. Graph Encoding

We consider graphs over finite vertex sets, with undirected edges or directed arcs, and finite sets of vertex and edge labels. Vertices and edges may be associated with multiple labels. We leave the encoding of directed arcs to the extended version of this paper.

$\mathcal{V}$	Finite set of vertices
$\mathcal{E} \subseteq (\mathcal{V} \times \mathcal{V})$	Finite set of edges
$\mathcal{G} = (\mathcal{V}, \mathcal{E}, t_\mathcal{V}, t_\mathcal{E})$	Graph
$\mathcal{L}_\mathcal{V}, \mathcal{L}_\mathcal{E}$	Finite sets labels
$f_\mathcal{V} : \mathcal{V} \to \mathcal{P}(\mathcal{L}_\mathcal{V})$	labels of a given vertex
$f_\mathcal{E} : \mathcal{E} \to \mathcal{P}(\mathcal{L}_\mathcal{E})$	labels of a given edge
$n = \lvert\mathcal{V}\rvert, m = \lvert\mathcal{E}\rvert$	number of vertices and edges

For each vertex i in $\mathcal{V}$, we introduce a vertex identifier, a prime e_i, which represents this vertex in credential and proofs. The symbol $\perp$, associated with identifier $e_\perp$ represents that a vertex is not present. All vertex identifiers are pair-wise different. We call the set of all vertex identifiers $\Xi_\mathcal{V}$, their product $\chi_\mathcal{V} = \Pi\Xi_\mathcal{V}$. For each label k in the label sets $\mathcal{L}_\mathcal{V}$ and in $\mathcal{L}_\mathcal{E}$, we introduce a prime representative e_k. All label representatives are pair-wise

Table I
PROOF OF KNOWLEDGE PREDICATES FOR GRAPH SIGNATURES.

Predicate	Description	
$\mathsf{possession}(\mathcal{G}, \sigma, \mu_i)$	Proof of possession of a graph signature σ	§V-B
$\mathsf{vertices}(\mathcal{G}, \varepsilon_i, \gamma_i)$	Proof of composition of graph vertices of a proof of possession	§V-B
$\mathsf{edges}(\mathcal{G}, \varepsilon_i, \varepsilon_i, \gamma_{(i,j)})$	Proof of composition of graph edges of a proof of possession	§V-B
$\mathsf{graph}(\mathcal{G}, \mu_i)$	Proof of representation and well-formedness	§V-B
$\mathsf{set}(\mathcal{V}, V)$	Representation of a set $V \subseteq \mathcal{V}$	§VI-A
$\mathsf{cover}(\mathcal{V}, V_1, \ldots, V_k)$	Vertex set coverage $\bigcup(V_1, \ldots, V_k) = \mathcal{V}$	§VI-A
$\mathsf{disjoint}(\mathcal{V}, V_1, \ldots, V_k)$	Vertex set pair-wise disjointness $\bigcap(V_1, \ldots, V_k) = \emptyset$	§VI-A
$\mathsf{partition}(\mathcal{V}, V_1, \ldots, V_k)$	Vertex set partition $\bigcup(V_1, \ldots, V_k) = \mathcal{V} \wedge \bigcap(V_1, \ldots, V_k) = \emptyset$	§VI-A
$\mathsf{edge}(\mathcal{G}, i, j)$	*VALID*-rule: Adjacency of (i, j)	§VI-B
$\mathsf{connected}(\mathcal{G}, i, j, \ell)$	*VALID*-rule: Existence of an ℓ-path between vertex i and vertex j	§VI-C
$\mathsf{isolated}(\mathcal{G}, i, j)$	*VALID*-rule: Isolation of vertices i and j: There exists no path between i and j	§VI-D

different. We call the set of all label representatives $\Xi_{\mathcal{L}}$, their product $\chi_{\mathcal{L}} = \Pi\Xi_{\mathcal{L}}$. Vertex identifiers and label representatives are disjoint:

$$\Xi_{\mathcal{V}} \cap \Xi_{\mathcal{L}} = \emptyset \quad \Leftrightarrow \quad \mathsf{gcd}(\chi_{\mathcal{V}}, \chi_{\mathcal{L}}) = 1.$$

A. Random Base Association

We encode vertices and edges into the exponents of integer commitments and CL-Signatures and make them therefore accessible to proofs of linear equations over exponents. We randomize the base association to vertices and edges: For a vertex index set $\mathcal{V} = \{0, \ldots, n-1\}$ with vertex identifiers e_i, we choose a uniformly random permutation $\pi_{\mathcal{V}}$ of set $\mathcal{V}$ to determine the base $R_{\pi(i)}$ to encode vertex i. Edge bases $R_{\pi(i,j)}$ are chosen analogously with a random permutation $\pi_{\mathcal{E}}$.

B. Encoding Vertices

To encode a vertex and its associated labels into a graph commitment or CL-Signature, we encode the product of the vertex identifier $e_i \in \Xi_{\mathcal{V}}$ and the prime representatives $e_k \in \Xi_{\mathcal{L}}$ for $k \in f_{\mathcal{V}}(i)$ of the labels into a single of the signature message. The product of prime representatives is encoded as exponent of dedicated vertex bases $R \in G_{\mathcal{V}}$.

C. Encoding Edges

To get a compact encoding and efficient proofs thereon, the encoding needs to maintain the graph structure and to allow us to access it to proof higher-level properties, such as connectivity and isolation. The proposal we make in this paper after evaluating multiple approaches is to use divisibility and coprimality similar to the CG-Encoding to afford us these efficient operations over the graph structure, while offering a compact encoding of edges.

1) Undirected Edge Encoding: Recall that each vertex is certified with an vertex identifier from $\Xi_{\mathcal{V}}$, e.g., e_i or e_j. For each edge $(i, j) \in \mathcal{E}$, we include an edge attribute as exponent of a random edge base $R_{\pi(i,j)} \in G_{\mathcal{E}}$, containing the product of the vertex identifiers and the associated label representatives $e_k \in \Xi_{\mathcal{L}}$ for $k \in f_{\mathcal{E}}(i, j)$ of the edge:

$$E_{(i,j)} := e_i \cdot e_j \cdot \Pi_{k \in f_{\mathcal{E}}(i,j)} e_k.$$

Whereas we usually consider simple graphs, specialities such as multigraphs, loops (i, i) encoded as e_i^2 or half-edges encoded as $(e_j, e_\perp)$ can be included.

2) Directed Edge Encoding: We call a directed edge an *arc*. For each arc (i, j), with tail i and head j, we include an edge attribute as exponent of a random edge base $R_{(i,j)} \in G_{\mathcal{E}}$, containing the product of the tail's vertex identifier and the the *square* of the head's vertex identifier (along with the label representatives $e_k \Xi_{\mathcal{L}}$ for $k \in f_{\mathcal{E}}(i, j)$):

$$E_{(i,j)} := e_i \cdot e_j^2 \cdot \Pi_{k \in f_{\mathcal{E}}(i,j)} e_k.$$

Whereas we usually consider simple graphs, specialities such as directed multigraphs, loops (i,i) encoded as e_i^3 or half-arcs encoded as $(e_j, e_\perp^2)$ or $(e_\perp, e_i^2)$ can be included.

D. Well-formed Graphs

Definition 1 (Well-formed graph): We call a graph encoding *well-formed* iff 1. the encoding only contains prime representatives $e \in \Xi_\mathcal{V} \cup \Xi_\mathcal{L}$ in the exponents of designated vertex and edge bases $R \in G_\mathcal{V} \cup G_\mathcal{E}$, 2. each vertex base $R \in G_\mathcal{V}$ contains exactly one vertex identifier $e_i \in \Xi_\mathcal{V}$, pair-wise different from other vertex identifiers and zero or more label representatives $e_k \in \Xi_\mathcal{L}$, and 3. each edge base $R \in G_\mathcal{E}$ contains exactly two vertex identifiers $e_i, e_i \in \Xi_\mathcal{V}$ and zero or more label representatives $e_k \in \Xi_\mathcal{L}$.

Theorem 4.1 (Unambiguous encoding): A well-formed graph encoding on the integers is unambiguous modulo the base association. [Proof in extended version]

V. Signatures on Committed Graphs

CL-signatures are signatures on committed messages, where messages can be contributed by issuer and user. This translates to a user committing to a hidden partial graph, which is then completed by the issuer. We establish the setup for the construction first, explain the proof of representation second, and the issuing third.

As a point of reference, we give the structure of the graph signatures first. We have bases $R_{\pi(i)} \in G_\mathcal{V}$, which store attributes encoding vertices, and bases $R_{\pi(i,j)} \in G_\mathcal{E}$, which store attributes encoding edges. The base association is randomized by permutations $\pi_\mathcal{V}$ and $\pi_\mathcal{E}$. The following congruence holds in $\mathbb{Z}_n^*$:

$$Z \equiv \pm \underbrace{R_{\pi(i)}^{e_i \Pi_{k \in f_\mathcal{V}(i)} e_k}}_{\forall \text{ vertices } i} \cdots \underbrace{R_{\pi(i,j)}^{e_i e_j \Pi_{k \in f_\mathcal{E}(i,j)} e_k}}_{\forall \text{ edges } (i,j)} A^e S^v$$

A. Setup and Key Generation

We define the setup and algorithm $\mathsf{Keygen}(1^\ell, Params)$ step by step. The setup builds on the CL-Signature setup as described in §II-D, where we need to establish additional elements, to prepare for set membership proofs. As part of the CL-Signature setup, the issuer chooses, uniformly at random, message-space bases $R_0, \ldots, R_{L-1} \in \mathrm{QR}_N$ and designates the bases $R_0, \ldots, R_{m-1}$ for vertex messages and bases $R_m, \ldots, R_{L-1}$ for edge messages.

In addition, the issuer chooses, uniformly at random, $R_\mathcal{V}, R_\mathcal{L} \in \mathrm{QR}_N$, pair-wise different from the message-space bases $R_0, \ldots, R_{L-1}$.

The issuer certifies the prime representatives of vertices $e_i \in \Xi_\mathcal{V}$ and labels $e_j \in \Xi_\mathcal{L}$, by CL-signing these primes: On input of a vertex representative e_i, the issuer chooses a random prime number e of length $\ell_e > \ell_\mathcal{M} + 2$, and a random number v of length $\ell_v = \ell_n + \ell_\mathcal{M} + \ell_r$. It computes

$$A = \left(\frac{Z}{R_\mathcal{V}^{e_i} S^v} \right)^{1/e} \bmod N$$

and outputs $\sigma_i = (A, e, v)$ and e_i. Label representatives are signed analoguously using base $R_\mathcal{L}$.

The issuer computes the products of vertex and label representatives $\chi_\mathcal{V}$ and $\chi_\mathcal{L}$ and an Integer commitments thereon:

$$C_\mathcal{V} = R^{\chi_\mathcal{V}} S^{r_\mathcal{V}} \bmod N \text{ and } C_\mathcal{L} = R^{\chi_\mathcal{L}} S^{r_\mathcal{L}} \bmod N.$$

The issuer creates a signature proof of knowledge showing that $\Xi_\mathcal{V}$ and $\Xi_\mathcal{L}$ are disjoint and that the products consist of all certified prime representatives for vertices and labels respectively.

$$\begin{aligned}
\sigma_\Xi = & \mathit{SPK}\{(\rho_1, \rho_2, (\forall i : \varepsilon_i, \nu_i'), \\
& \qquad (\forall k : \varepsilon_k, \nu_k'), \alpha, \beta, \rho') : \pmod N \\
& C_\mathcal{V} \equiv \pm R^{\Pi \Xi_\mathcal{V}} S^{\rho_1} \ \wedge \\
& C_\mathcal{L} \equiv \pm R^{\Pi \Xi_\mathcal{L}} S^{\rho_2} \ \wedge \\
& \forall e_i \in \Xi_\mathcal{V} : Z \equiv \pm R_\mathcal{V}^{e_i} A'^{\varepsilon_i} S^{\nu_i'} \ \wedge \\
& \forall e_k \in \Xi_\mathcal{L} : Z \equiv \pm R_\mathcal{L}^{e_k} A'^{\varepsilon_k} S^{\nu_k'} \ \wedge \\
& R \equiv \pm C_\mathcal{V}^\alpha C_\mathcal{L}^\beta S^{\rho'} \ \wedge \\
& \forall e_i, e_k \in \pm\{0,1\}^{\ell_\mathcal{M}} \ \wedge \\
& \forall \varepsilon_i, \varepsilon_k \in [2^{\ell_e - 1} + 1, 2^{\ell_e} - 1]\}
\end{aligned}$$

Finally, Keygen outputs as public key pk_I an extension of the CL-signature public key, including the certified prime representatives for vertices and labels:

$$pk_\mathsf{I} = (N, R_0, \ldots, R_{L-1}, R_\mathcal{V}, R_\mathcal{L}, S, Z, \Xi_\mathcal{V}, \Xi_\mathcal{L}, \sigma_\Xi)$$

The private key sk_{I} is the factorization of the issuer's special RSA modulus N.

B. Proof of Representation

For a full proof of representation, we need to establish that the encoded graph in a graph commitment or CL-Signature is indeed well-formed (Def. 1). We represent this proof by predicate $\mathsf{graph}(\mathcal{G})$. Given a graph commitment C the prover and verifier engage in the following proof of representation (the proof for a CL credential work analoguously). We show that vertex bases contain a bi-partition of one and only one vertex identifier $e_i \in \Xi_{\mathcal{V}}$ and a set of labels $e_l \in \Xi_{\mathcal{L}}$. Edge bases contain a bi-partition of a product of exactly two vertex identifiers $(e_i \cdot e_j)$ and a set of labels $e_l \in \Xi_{\mathcal{L}}$. To prove that the representation contains exactly one vertex identifier for a vertex base and two vertex identifiers for an edge base, we establish a set membership proof.

1. Commitments The prover computes Integer commitments on the exponents of all vertex and edge bases. First, the prover computes commitments on all messages to allow their decomposition into components. These are the commitments $\mathsf{Commit}(\mathsf{possession}(\mathcal{G}, \sigma, \mu_i); r_i, r_{(i,j)})$ in $\mathbb{Z}_N^*$ with uniformly chosen randomness $r_i, r_{(i,j)} \in \{0,1\}^\ell$:

$$C_i = R^{e_i \Pi_{k \in f_{\mathcal{V}}(i)} e_k} S^{r_i}$$
$$C_{(i,j)} = R^{e_i e_j \Pi_{k \in f_{\mathcal{E}}(i,j)} e_k} S^{r_{(i,j)}}$$

For each vertex i, the prover computes commitments $\mathsf{Commmit}(\mathsf{vertices}(\mathcal{G}); \breve{r}_i)$ in $\mathbb{Z}_N^*$ on vertex attribute and identifier using uniformly-chosen randomness $\breve{r}_i \in \{0,1\}^\ell$:

$$\breve{C}_i = R^{e_i} S^{\breve{r}_i}.$$

For edges (i,j), the commitments are in $\mathbb{Z}_N^*$ with uniformly chosen randomness $\breve{r}_{(i,j)}, \dot{r}_{(i,j)} \in \{0,1\}^\ell$. We call them $\mathsf{Commit}(\mathsf{edges}(\mathcal{G}); \breve{r}_{(i,j)}, \dot{r}_{(i,j)})$:

$$\breve{C}_{(i,j)} = R^{e_i e_j} S^{\breve{r}_{(i,j)}} \quad \text{and} \quad \dot{C}_i = R^{e_i} S^{\dot{r}_{(i,j)}}.$$

2. Proof of knowledge. We construct the proof of possession and well-formedness step by step, where it is understood the proofs will be done in one compound proof of knowledge with *referential integrity between the secret exponents*. Let us consider a proof fragment for vertices i, j and an edge (i,j) committed in a graph commitment C (the same proof structure is used for CL-Signatures).

2.1 Proof of representation. We prove that commitment C can be decomposed into commitments C_i, C_j, one for each vertex i, j and one commitment $C_{(i,j)}$ for each edge (i,j):

$$PK\{(\mu_i, \mu_j, \mu_{(i,j)}, \rho, \rho_i, \rho_j, \rho_{(i,j)}) : \pmod N$$
$$C \equiv \pm \prod_{i,j} R_{\pi(i)}^{\mu_i} R_{\pi(j)}^{\mu_j} \prod_{(i,j)} R_{\pi(i,j)}^{\mu_{(i,j)}} S^{\rho} \quad \wedge \quad (1)$$
$$C_i \equiv \pm R^{\mu_i} S^{\rho_i} \quad \wedge \quad C_j \equiv \pm R^{\mu_j} S^{\rho_j} \quad \wedge \quad (2)$$
$$C_{(i,j)} \equiv \pm R^{\mu_{(i,j)}} S^{\rho_{(i,j)}}\}. \quad (3)$$

The same proof of representation can be applied to graph signatures to prove the predicate $\mathsf{possession}$:

$$Z \equiv \pm \prod_{i,j} R_{\pi(i)}^{\mu_i} R_{\pi(j)}^{\mu_j} \prod_{(i,j)} R_{\pi(i,j)}^{\mu_{(i,j)}} A'^{\varepsilon'} S^{\rho'}$$

2.2 Vertex composition. Second, we need to show properties of the vertex composition that the encoding for each vertex i contains exactly one vertex identifier $e_i \in \Xi_{\mathcal{V}}$ and zero or multiple label representatives $e_k \in \Xi_{\mathcal{L}}$. We show this structure with help of the commitments $\breve{C}_i$ and set membership and prime-encoding OR proofs. This proof is executed for all vertices.

$$PK\{(\forall i : \varepsilon_i, \breve{\rho}_i, \gamma_i, \rho_i') : \pmod N$$
$$\breve{C}_i \equiv \pm R^{\varepsilon_i} S^{\breve{\rho}_i} \quad \wedge \quad C_i \equiv \pm \breve{C}_i^{\gamma_i} S^{\rho_i'} \quad \wedge \quad (4)$$
$$\gamma_i[C_i] \subseteq \Xi_{\mathcal{L}} \quad \wedge \quad \varepsilon_i[\breve{C}_i] \in \Xi_{\mathcal{V}}\}. \quad (5)$$

Clause 4 establishes the predicate $\mathsf{vertices}(\mathcal{G}, \mu_i, \varepsilon_i, \gamma_i)$, where the second term links to the possession commitments.

2.3 Edge composition. Third, we prove the structure of each edge (i,j) over the commitments $C_{(i,j)}$, showing that each commitment contains exactly two vertex identifiers $e_i, e_j \in \Xi_{\mathcal{V}}$ as well as zero or more label representative $e_k \in \Xi_{\mathcal{L}}$:

$$PK\{(\varepsilon_j, \rho_{(i,j)}, \gamma_{(i,j)}, \rho'_{(i,j)}) : \pmod N$$
$$\breve{C}_{(i,j)} \equiv \pm \dot{C}_i^{\varepsilon_j} S^{\rho_{(i,j)}} \quad \wedge \quad (6)$$
$$C_{(i,j)} \equiv \pm \breve{C}_{(i,j)}^{\gamma_{(i,j)}} S^{\rho'_{(i,j)}} \quad \wedge \quad (7)$$
$$\gamma_{i,j} \subseteq \Xi_{\mathcal{L}}\}. \quad (8)$$

Clauses 6 and 7 establish the predicate $\mathsf{edges}(\mathcal{G}, \varepsilon_i, \varepsilon_j, \gamma_{(i,j)})$, where Clause 7 binds the edges commitments to the possession commitments.

2.4 Pair-wise difference. Finally, we prove pair-wise difference of vertices by showing that the vertex representatives are pair-wise co-prime over the commitments $\breve{C}_i$ and $\breve{C}_j$.

$$PK\{(\forall i,j : \alpha_{i,j}, \beta_{i,j}, \rho_{i,j}) : \\ R \equiv \pm \breve{C}_i^{\alpha_{i,j}} \breve{C}_j^{\beta_{i,j}} S^{\rho_{i,j}} \pmod N\}. \quad (9)$$

The compound proof of knowledge establishes the predicate $\mathsf{graph}(\mathcal{G}, \mu_i)$.

C. Joint Graph Issuing

To jointly issue a graph CL-signature, a user commits to a hidden partial graph and the issuer adds further elements to the graph (cf. Section II-D)

In the setup, the issuer establishes a user vertex space and issuer vertex space, i.e., a bi-partition on vertex and edge bases, $G_{\mathcal{V}}$ and $G_{\mathcal{E}}$ and on vertex identifiers $\Xi_{\mathcal{V}}$. Thus, user and issuer can encode partial graphs without interfering with each other.

In the joint graph issuing, user and issuer designate and disclose connection points (vertex identifiers) that allow the user and the issuer to connect their sub-graphs deliberately. The user constructs a graph representation by choosing two uniformly random permutation $\pi_{\mathcal{V}}$ and $\pi_{\mathcal{E}}$ for the base association on the user bases and commits to his sub-graph in a graph commitment. The user interacts with the issuer in a proof of representation of his committed sub-graph. The issuer verifies this proof, chooses uniformly random permutations for his graph elements and encodes them into his base range. The issuer creates the pre-signature of the CL-Signature scheme on the entire graph, proving that the added sub-graph is well-formed. The user completes the CL-Signature with his own randomness.

VI. Proofs of Graph Properties

Having established encoding and foundational bootstrapping cycle of proof of representation and issuing, we continue to establish a library of graph proof predicates. First, we explore with proofs over vertex and edge sets, including coverage and pair-wise disjointness, which are the basis of partition proofs. Second, we discuss different proofs over presence and absence of labels. Third, we establish results on connectivity and isolation in undirected and directed graphs.

A. Sets of Vertices and Cumulative Products

Arguing over sets of vertices of hidden graphs is a basic tool for graph proofs.

1) Sets as Products: We represent a set of vertices $V \subseteq \mathcal{V}$ with ℓ elements by the cumulative product of its vertex identifiers:

$$E_V = \Pi_{i \in V} e_i.$$

The normal set representation only includes the vertex identifiers of the vertex set, the extended set representation also all associated vertex labels. We will use proofs over cumulative products repeatedly an establish a generic interface for those. In the following, we rename the vertex identifier indices to range over $1, \ldots, \ell$, wlog.

1. Commitments. The prover commits to the to the vertex set representation as the cumulative product E_V. To prepare the proof of representation the prover establishes intermediate commitments $\mathsf{Commit}(\mathsf{set}(\mathcal{V}, V); \breve{r}_1, \ldots, \breve{r}_\ell)$ in $\mathbb{Z}_N^*$ on partial products using uniformly chosen randomness $\breve{r}_1, \ldots, \breve{r}_\ell$:

$$\breve{C}_{V,1} = R^{e_1} S^{\breve{r}_1}, \ldots, \breve{C}_{V,\ell} = R^{\Pi_1^\ell e_i} S^{\breve{r}_\ell}.$$

2. Proof of Representation. To establish a proof of representation of a set $\mathsf{set}(\mathcal{V}, V)$ that a cumulative product E_V is composed of the identifiers of certified vertices, the prover engages with the verifier in the following proof of knowledge over the cumulative product:

$$PK\{((\forall i : \mu_i, \varepsilon_i), \breve{\rho}_1, \ldots, \breve{\rho}_\ell) : \pmod N \\ \mathsf{possession}(\mathcal{G}, \sigma, \mu_i) \ \wedge \ \mathsf{vertices}(\mathcal{G}, \mu_i, \varepsilon_i) \\ \breve{C}_{V,1} \equiv \pm R^{\varepsilon_1} S^{\breve{\rho}_1} \ \wedge \\ \breve{C}_{V,2} \equiv \pm \breve{C}_{V,1}^{\varepsilon_2} S^{\breve{\rho}_2} \ \wedge \ldots \\ \breve{C}_{V,\ell} \equiv \pm \breve{C}_{V,\ell-1}^{\varepsilon_\ell} S^{\rho_\ell}$$

Remark 2 (Edge Sets): Edge sets can be represented as products of their vertex identifiers, as well. This is a degenerate representation as it does not maintain the the edge structure, but only the vertices

present, however as we shall see in Section VI-C it serves its purpose for edge partitions.

2) Coverage: The predicate cover establishes $\bigcup\{V_1, V_2, \ldots, V_k\} = \mathcal{V}$, that is, a given set of vertex sub-sets complete covers the graph vertices. In terms of the divisibility proofs, this equivalent to the product of all graph vertex identifiers $\chi_{\mathcal{V}}$ dividing the product representation of the sets:

$$\chi_{\mathcal{V}} | \prod_{i=1}^{k} E_{V_i} \Leftrightarrow \exists a : a\chi_{\mathcal{V}} = \prod_{i=1}^{k} E_{V_i},$$

where $E_{V_i} = \prod_{j \in V_i} e_j$. The product representation of $\Xi_{\mathcal{V}}$ is signed with $C_{\mathcal{V}}$ as part of the issuing. Thus, given proofs for $\mathsf{set}(\mathcal{G}, V_i)$, we compute commitments and proofs for $\mathsf{set}(\mathcal{G}, \{V_1, \ldots, V_k\})$, which results in a commitment on the cumulative product of all sets: $C_{\bar{V}} = R^{\prod_{i=1}^{k} E_{V_i}} S^{\bar{r}}$.

To complete the proof of coverage, the prover and the verifier engage in the following proof of knowledge:

$$\begin{aligned} PK\{(\alpha, \rho) : & \\ & \mathsf{set}(\mathcal{G}, V_1) \quad \wedge \cdots \wedge \quad \mathsf{set}(\mathcal{G}, V_k) \quad \wedge \\ & \mathsf{set}(\mathcal{G}, \{V_1, \ldots, V_k\}) \quad \wedge \\ & C_{\bar{V}} \equiv \pm C_{\mathcal{V}}^{\alpha} S^{\rho} \pmod N \end{aligned}$$

3) Pair-wise Disjointness: The predicate disjoint establishes that vertex sets do not have a joint vertex, $\bigcap\{V_1, V_2, \ldots, V_k\} = \emptyset$. We use the fact that two vertex sets $V_i \subseteq V$ and $V_j \subseteq \mathcal{V}$ are pair-wise disjoint if their product representations are coprime:

$$V_i \cup V_j \quad \Leftrightarrow \quad \mathsf{gcd}(E_{V_i}, E_{V_j}) = 1.$$

Based on given commitments C_{V_i} and proofs for predicates $\mathsf{set}(\mathcal{G}, V_i)$ and $\mathsf{set}(\mathcal{G}, V_j)$, we can establish co-primality and thereby disjointness as follows:

$$\begin{aligned} PK\{(\forall i, j : \alpha_{i,j}, \beta_{i,j}, \rho_{i,j}) : \quad & \pmod N \\ \mathsf{set}(\mathcal{G}, V_1) \quad \wedge \cdots \wedge \quad & \mathsf{set}(\mathcal{G}, V_k) \quad \wedge \\ \forall i, j : R \equiv \pm C_{V_i}^{\alpha_{i,j}} C_{V_j}^{\beta_{i,j}} S^{\rho_{i,j}} & \end{aligned}$$

4) Partition: For a partition proof, we compose coverage and pair-wise disjointness operating with given commitments and proofs for predicates $\mathsf{set}(\mathcal{G}, V_i)$:

$$\bigcup\{V_1, V_2, \ldots, V_k\} = \mathcal{V} \wedge \bigcap\{V_1, V_2, \ldots, V_k\} = \emptyset.$$

B. Arguing over Edges

To prove the edge predicate, that is, to show that an edge base with an exponent E encodes the edge (i, j), we need to show that

$$(e_i e_j) | E \Leftrightarrow \exists a : a(e_i e_j) = E.$$

This is realized by the prover engaging with the verifier in the following proof of knowledge:

$$\begin{aligned} \textit{PK}\{((\forall i : \mu_i), \mu', \rho') : & \\ & \mathsf{possession}(\mathcal{G}, \mu_i) \quad \wedge \\ & C_{(i,j)} \equiv \pm (R^{e_i e_j})^{\mu'} S^{\rho'} \pmod N\}. \end{aligned}$$

Negated *VALID* predicates, such as $\neg\mathsf{edge}(i, j)$ stating that there is no edge (i, j) in a graph's edge set, are hard to prove in zero-knowledge proofs of knowledge, because the verifier needs to be convinced that the prover did not withhold information. We therefore need to show for all edges in $\mathcal{E}$ that none of them is (i, j). We reduce $\neg\mathsf{edge}(i, j)$ to $\mathsf{isolated}(\mathsf{i}, \mathsf{j})$ and to use the proof constructed in Section VI-C.

C. Connectivity

A major area of interest for hidden-graph proofs is connectivity: How can we show that different vertices of a hidden graph are connected by a chain of ℓ edges?

Wlog., let end vertices be i and j. The predicate $\mathsf{connected}(\mathcal{G}, i, j, \ell)$ means that there exists a sequence of at most ℓ edges $(i, k_1), \ldots, (k_n, j)$ such that the end vertex of one edge is the start vertex of the next.

1) Undirected Graph:

Example 1: Let us consider the following chain of connected edges:

$$(e_i \cdot e_1), (e_1 \cdot e_2), (e_2 \cdot e_3), (e_3 \cdot e_j)$$

We observe that two vertices are connected if and only if there exits a sequence of edge products, such that their the edges match in the joint vertex

identifier. By that, we have for instance for the first pair of edges:

$$e_1|(e_i \cdot e_1) \wedge e_1|(e_1 \cdot e_2)$$

Therefore, we can express connectivity by a chained proof of divisibility constructed from the commitments and proofs of the predicate $\mathsf{edges}(\mathcal{G}, \varepsilon_i, \varepsilon_j, \gamma_{(i,j)})$. Recall that this predicate creates commitments in $\mathbb{Z}_N^*$:

$$\breve{C}_{(i,j)} = R^{e_i e_j} S^{\breve{r}_{(i,j)}} \quad \text{and} \quad \dot{C}_i = R^{e_i} S^{\dot{r}_{(i,j)}}$$

for each edge (i, j), allowing us to compose statements adjacent edges via $\breve{C}_{(i,j)}$ and the subsequent $\dot{C}_j$. We show that a joint factor ε divides adjacent edges (i, j) and (j, k) with the following proof of knowledge:

$$\begin{aligned} PK\{(\varepsilon, \rho, \rho') : \quad & (\mathrm{mod}\ N) \\ & \breve{C}_{(i,j)} \equiv \pm \dot{C}_i^{\varepsilon} R^{\rho} \quad \wedge \quad \dot{C}_j = R^{\varepsilon} S^{\rho'} \}. \end{aligned}$$

2) Directed Graphs: We observe that the matching of arc products is a necessary condition for connectivity in directed graphs. It is not sufficient because arc directions need to match, too. Reachability in directed graphs is not symmetric. Let us consider the sequence of arcs: $i \curvearrowright 1, 1 \curvearrowright 2, 2 \curvearrowright 3, 3 \curvearrowright j$, where the encoding denotes the tip of the arc by encoding e_i^2. Thus, in this case the divisibility proof is:

$$e_1^2|(e_i \cdot e_1^2) \wedge e_1|(e_1 \cdot e_2).$$

Based on the assumption that this issuer only certifies well-formed graphs (in which each arc has only one tip), such a chain of divisibility proofs convinces the verifier of the connected predicate analogously to the undirected case.

D. Isolation

For vertices i and j $\mathsf{isolated}(i, j) = \neg\mathsf{connected}(i, j)$ means that there exits no connected path between both vertices i and j.

1) Undirected Graph: For undirected graphs, isolation means that the vertices i and j are in separate sub-graphs.

Two vertices i and j are isolated, if there exits a bi-partition of the edge set $V' \cup V'' = \mathcal{E} \wedge V' \cup V'' = \emptyset$, such that wlog. $i \in V'$ and $j \in V''$. Recall that Section VI-A represents an edge set in a degenerate form, as the product of the edges' vertex identifiers E' and E''. We obtain commitments and proofs for the predicates $\mathsf{set}(\mathcal{G}, V')$ and $\mathsf{set}(\mathcal{G}, V'')$ which give us two commitments in $\mathbb{Z}_N^*$:

$$\breve{C}_{E'} = R^{E'} S^{\breve{r}'} \quad \text{and } \breve{C}_{E''} = R^{E''} S^{\breve{r}''}.$$

We can derive coverage already from the fact that all commitments of the predicate edges are used to establish the cumulative products for both edge sets. The disjointness of the edge-set bi-partition gives the isolation result, which we prove by showing that both products are coprime:

$$\mathsf{gcd}(E', E'') = 1 \quad \Leftrightarrow \quad \exists a, b : aE' + bE'' = 1.$$

The complete the proof of the predicate isolated, the prover and the verifier engage in the following proof of knowledge:

$$\begin{aligned} PK\{(\alpha, \beta, \rho) : \quad & (\mathrm{mod}\ N) \\ & \mathsf{set}(\mathcal{G}, V') \quad \wedge \quad \mathsf{set}(\mathcal{G}, V'') \quad \wedge \\ & R \equiv \pm \breve{C}_{E'}^{\alpha} \breve{C}_{E''}^{\beta} S^{\rho} \end{aligned}$$

Remark 3: The isolation predicate constructed in this section argues over the edges only. In actual topologies, such as infrastructure clouds, it is however the case that graph labels are important to decide upon connectivity and isolation. This holds in particular for the VLAN IDs of virtualized infrastructures, which allow communication if components have matching VLAN IDs. The isolation predicate can be easily extended to show pair-wise disjointness for labels as well, which we demonstrated in a companion paper for an infrastructure cloud example.

2) Directed Graphs: Isolation proofs in directed graphs need to account for arc direction and are not symmetric. We sketch the concept of an isolation proof for a directed graph briefly. We have isolation from vertex i to vertex j, if there exists a tri-partition of the graph arcs into three sets A, B and C with the following properties:

- A contains the arcs containing i.
- B is the boundary zone between both sub-graphs, of arcs between vertices of A and B.
- C contains the arcs containing j.

- ΠA and ΠC are coprime, i.e., they have no vertex identifier in common.
- i and j are coprime with respect to ΠB, i.e., the boundary zone does not contain i or j.
- For all arcs in B holds, arcs only go from C to A and not in the opposite direction.

The proofs of knowledge for this statement can be realized with the techniques demonstrated above.

VII. Efficiency Analysis

We display the efficiency analysis for the proof predicates in Table II, where each row shows the overhead over the basis predicate stated in the first column. We measure computational complexity in multi-base and modular exponentiations. The communication complexity is dominated by the transmitted group elements from $\mathbb{Z}_N^*$, which is equal to the number of multi-base exponentiations (one for each Integer and Schnorr proof commitment). The most expensive proof is the complete graph representation established in the issuing, where the set membership proofs (4 MExps) and the OR-based subset proofs (6 MExps) constitute significant overhead. In the down-stream proofs, the verifier trusts the issuer to only certify well-formed graphs.

The modular exponentiations for message bases R_i are with small exponents of size of $\ell_{\mathcal{M}} \ll \ell_N$, where the parameter $\ell_{\mathcal{M}}$ can be chosen similarly small as in Direct Anonymous Attestation (DAA) [3]. Implementing the proofs of representation as multi-base exponentiations will reduce the number of multiplications significantly and thereby offer a significant speed-up. Following the analysis of Camenisch and Groth [23], we can estimate that a multi-base exponentiation takes about 10% more time than a single modular exponentiation with same exponent size.

In addition, the Σ-proofs employed in this work benefit from batch-proof techniques, such as [24]. The graph proofs are likely to be transformed to signature proofs of knowledge with the Fiat-Shamir heuristic [17] and can thereby be computed offline.

VIII. Related Work

Zero-knowledge proofs on graphs and their properties is a classic area of research and have been instrumental in showing that there exist zero-knowledge proof systems for all NP languages, e.g., [25], [26] Both proofs use a metaphor of locked boxes to construct known-graph proofs of Graph 3-Colorability (G3C) or Directed Hamiltonian cycles (DHC). The constructions focus on zero-knowledge proofs of knowledge and do not cater for a level of indirection through a signature scheme or proofs of knowledge on graph properties in a hidden-graph setting.

A related notion to full graph signatures is transitive or homomorphic signature schemes, such as [27], [28], [29]. They are concerned with the transitive closure of signatures on graph elements, such that from signatures from edges (i, j) and (j, k) everybody can derive a valid signature on the edge (i, k). These signature schemes have the advantage that one can produce a signature of the transitive path over multiple edges. Therefore, they allow to show signatures equivalent to the connected predicate without disclosing the number of edges on the path and without overhead because of path length. The constructions are not meant to be on committed graphs and consider the signatures as public information. They have limited support for labels and do not have provisions for proofs of isolation as signatures could be withheld.

IX. Conclusion and Future Work

We have introduced a signature scheme on committed graphs together with a library of proof predicates on sets, connectivity and isolation. The scheme covers undirected and directed, unlabeled and labeled graphs and enables honest-verifier zero-knowledge proofs of knowledge over graph properties, while keeping the graph itself confidential. It constitutes a building block to overcome the requirement gap between the confidentiality requirements of a provider and the integrity requirements of a tenant.

The signature scheme and its proofs are created in a special RSA setting; their security is based on the Strong RSA assumption. The signature scheme is based on the Camenisch-Lysyanskaya (CL) signature scheme [2] and its existential unforgeability directly derived from that. The proofs can be

Table II
EFFICIENCY OF PROOFS OF PREDICATES IN MULTI-BASE AND MODULAR EXPONENTIATIONS (MULTIEXPS AND MODEXPS). FOR A SIMPLE GRAPH HOLDS $m \leq \frac{n(n-1)}{2}$. IN MOST CASES IN PRACTICE HOLDS $k \ll \ell \leq n$ AND $O(k\ell) = O(n)$.

Predicate	**Basis**	**Commitments**	**MultiExps**	**ModExps**	
		#	#	#	O
$\mathsf{possession}(\mathcal{G}, \sigma, \mu_i)$		$n+m$	$2n+2m+1$	$5n+5m+2$	$O(n+m)$
$\mathsf{vertices}(\mathcal{G})$	possession	n	$3n$	$6n$	$O(n)$
$\mathsf{edges}(\mathcal{G})$	possession	$2m$	$4m$	$8m$	$O(m)$
$\mathsf{graph}(\mathcal{G})$		$2n+3m$	$n^2+8n+8m+1$	$3n^2+21n+19m+1$	$O(n^2)$
$\mathsf{set}(\mathcal{V}, V)$, with $\ell = \lvert V \rvert$	vertices	ℓ	2ℓ	4ℓ	$O(\ell) = O(n)$
$\mathsf{cover}(\mathcal{V}, V_1, \ldots, V_k)$	vertices	$k(\ell+1)$	$2k(\ell+1)+1$	$4k(\ell+1)+2$	$O(k\ell)$
$\mathsf{disjoint}(\mathcal{V}, V_1, \ldots, V_k)$	vertices	$k\ell$	$k^2+2k\ell$	$3k^2+4k\ell$	$O(k^2+k\ell)$
$\mathsf{partition}(\mathcal{V}, V_1, \ldots, V_k)$	vertices	$k(\ell+1)$	$k^2+2k(\ell+1)$	$3k^2+4k(\ell+1)+2$	$O(k^2+k\ell)$
$\mathsf{edge}(\mathcal{G}, i, j)$	possession	0	1	2	$O(1)$
$\mathsf{connected}(\mathcal{G}, i, j, \ell)$	edges	0	2ℓ	4ℓ	$O(\ell) = O(m)$
$\mathsf{isolated}(\mathcal{G}, i, j)$	edges	m	$2m+1$	$4m+3$	$O(m)$

transformed to signature proofs of knowledge with the Fiat-Shamir [17] heuristic, secure in the Random Oracle Model. The constructions for the proof predicates are efficient and practical.

As future work, we see great potential in linking the graph signatures to Direct Anonymous Attestation (DAA) [3]. This allows the combination of attestation results for system components (e.g., physical and virtual hosts) with statements over the system topology. A bridge between a formal cloud security assurance language, such as *VALID* [21], and predicates for graph proofs seems like an important step to link virtualized systems analysis and certification of topology structures.

REFERENCES

[1] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[2] J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," in *Security in Communication Networks, Third International Conference, SCN 2002*, ser. Lecture Notes in Computer Science, S. Cimato, C. Galdi, and G. Persiano, Eds., vol. 2576. Springer Verlag, 2003, pp. 268–289.

[3] E. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in *Proc. 11th ACM Conference on Computer and Communications Security*. acm press, 2004, pp. 225–234.

[4] C. P. Schnorr, "Efficient signature generation for smart cards," *Journal of Cryptology*, vol. 4, no. 3, pp. 239–252, 1991.

[5] I. Damgård and E. Fujisaki, "An integer commitment scheme based on groups with hidden order," in *Advances in Cryptology — ASIACRYPT 2002*, ser. Lecture Notes in Computer Science, vol. 2501. Springer, 2002.

[6] E. Fujisaki and T. Okamoto, "Statistical zero knowledge protocols to prove modular polynomial relations," in *Advances in Cryptology — CRYPTO '97*, ser. Lecture Notes in Computer Science, B. Kaliski, Ed., vol. 1294. Springer Verlag, 1997, pp. 16–30.

[7] D. Chaum and T. P. Pedersen, "Wallet databases with observers," in *Advances in Cryptology — CRYPTO '92*, ser. Lecture Notes in Computer Science, E. F. Brickell, Ed., vol. 740. Springer-Verlag, 1993, pp. 89–105.

[8] J. Camenisch and M. Michels, "Proving in zero-knowledge that a number n is the product of two safe primes," in *Advances in Cryptology — EUROCRYPT '99*, ser. Lecture Notes in Computer Science, J. Stern, Ed., vol. 1592. Springer Verlag, 1999, pp. 107–122.

[9] F. Boudot, "Efficient proofs that a committed number lies in an interval," in *Advances in Cryptology — EUROCRYPT 2000*, ser. Lecture Notes in Computer Science, B. Preneel, Ed., vol. 1807. Springer Verlag, 2000, pp. 431–444.

[10] A. Chan, Y. Frankel, and Y. Tsiounis, "Easy come – easy go divisible cash," in *Advances in Cryptology — EUROCRYPT '98*, ser. Lecture Notes in Computer Science, K. Nyberg, Ed., vol. 1403. Springer Verlag, 1998, pp. 561–575.

[11] N. Barić and B. Pfitzmann, "Collision-free accumulators and fail-stop signature schemes without trees," in *Advances in Cryptology — EUROCRYPT '97*, ser. Lecture Notes in Computer Science, W. Fumy, Ed., vol. 1233. Springer Verlag, 1997, pp. 480–494.

[12] I. Damgård and E. Fujisaki, "An integer commitment

scheme based on groups with hidden order," http://eprint.iacr.org/2001, 2001.
[13] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology – CRYPTO '91*, ser. Lecture Notes in Computer Science, J. Feigenbaum, Ed., vol. 576. Springer Verlag, 1992, pp. 129–140.
[14] S. Brands, "Rapid demonstration of linear relations connected by boolean operators," in *Advances in Cryptology — EUROCRYPT '97*, ser. Lecture Notes in Computer Science, W. Fumy, Ed., vol. 1233. Springer Verlag, 1997, pp. 318–333.
[15] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Advances in Cryptology — CRYPTO '94*, ser. Lecture Notes in Computer Science, Y. G. Desmedt, Ed., vol. 839. Springer Verlag, 1994, pp. 174–187.
[16] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," in *Advances in Cryptology — CRYPTO '97*, ser. Lecture Notes in Computer Science, B. Kaliski, Ed., vol. 1296. Springer Verlag, 1997, pp. 410–424.
[17] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology — CRYPTO '86*, ser. Lecture Notes in Computer Science, A. M. Odlyzko, Ed., vol. 263. Springer Verlag, 1987, pp. 186–194.
[18] S. Goldwasser, S. Micali, and R. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 281–308, Apr. 1988.
[19] J. Camenisch, R. Chaabouni, and a. shelat, "Efficient protocols for set membership and range proofs," in *Advances in Cryptology-ASIACRYPT 2008*. Springer, 2008, pp. 234–252.
[20] J. Camenisch and T. Groß, "Efficient attributes for anonymous credentials," *ACM Transactions on Information and System Security (TISSEC)*, vol. 15, no. 1, p. 4, 2012.
[21] S. Bleikertz and T. Groß, "A Virtualization Assurance Language for Isolation and Deployment," in *IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY'11)*. IEEE, Jun 2011.
[22] S. Bleikertz, T. Groß, and S. Mödersheim, "Automated Verification of Virtualized Infrastructures," in *ACM Cloud Computing Security Workshop (CCSW'11)*. ACM, Oct 2011.
[23] J. Camenisch and J. Groth, "Group signatures: Better efficiency and new theoretical aspects," in *proceedings of SCN '04*, ser. Lecture Notes in Computer Science, vol. 3352, 2004, pp. 120–133.
[24] K. Peng, C. Boyd, and E. Dawson, "Batch zero-knowledge proof and verification and its applications," *ACM Transactions on Information and System Security (TISSEC)*, vol. 10, no. 2, p. 6, 2007.
[25] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems," *Journal of the ACM (JACM)*, vol. 38, no. 3, pp. 690–728, 1991.
[26] M. Blum, "How to prove a theorem so no one else can claim it," in *Proceedings of the International Congress of Mathematicians*, vol. 1, 1986, p. 2.
[27] S. Micali and R. L. Rivest, "Transitive signature schemes," in *Topics in Cryptology-CT–RSA 2002*. Springer, 2002, pp. 236–243.
[28] R. Johnson, D. Molnar, D. Song, and D. Wagner, "Homomorphic signature schemes," in *Topics in Cryptology–CT-RSA 2002*. Springer, 2002, pp. 244–262.
[29] M. Bellare and G. Neven, "Transitive signatures based on factoring and rsa," in *Advances in Cryptology–ASIACRYPT 2002*. Springer, 2002, pp. 397–414.

APPENDIX

A. Proof Techniques for CL Set-Membership

We will need to prove that graph elements used in a user's committed graph are indeed certified by an issuer. To that end, we need to proof an exact set membership. Partially, the graph encoding used needs to be associated to certified semantics.

We obtain a set membership proof drawing on inspiration from the set membership work of Camenisch, Chaabouni and shelat [19]: Whereas their signature-based set-membership protocol is based on Boneh-Boyen signatures, we create a variant based on Strong-RSA CL-signatures with the same paradigm: 1. Publishing signatures on all set members, 2. blinding the signature for the set member to prove membership, and 3. proof of representation and equality of exponents. Given that the CL-Signature scheme is multi-use unlinkable, the published signatures can be used for multiple proofs.

Consider a set $\mathcal{S} = \{m_0, \ldots, m_i, \ldots, m_l\}$ The issuer signs all possible members m_i of the set in CL-Signatures $\sigma_i = (A, e, v)$ and publishes the set of signature-message pairs $\{(m_i, \sigma_i)\}$ integerly.

In the setup of a Strong-RSA CL-Signature scheme, the signature σ_i will fulfill the following equation:

$$Z \equiv \pm R_{\mathcal{S}}^{m_i} A^e S^v \pmod{N}$$

A prover shows that a value μ in a zero-knowledge proof, say a committed value $C = g^\mu h^r \bmod N$, is a member of the set $\mathcal{S}$ proceeds as follows. The prover randomizes the signature $\sigma_i = (A, e, v)$ of the corresponding message m_i:

$$(A' := AS^{-r} \bmod N, e, v' := v + er)$$

as in a standard proof of possession of a CL-Signature (cf. §II-D). The prover sends the randomized A' to the verifier and proves equality of exponents as follows:

$$\begin{aligned} PK\{&(\mu, \rho, \varepsilon, \nu') : \\ &C \equiv \pm g^\mu h^\rho \pmod{N} \ \wedge \\ &Z \equiv \pm R_{\mathcal{S}}^{\mu} A'^{\varepsilon} S^{\nu'} \pmod{N} \ \wedge \\ &\mu \in \pm\{0,1\}^{\ell_{\mathcal{M}}} \ \wedge \ \varepsilon \in [2^{\ell_e - 1} + 1, 2^{\ell_e} - 1]\} \end{aligned}$$

For a secret message μ in an Integer commitment C and a established signature distribution for set $\mathcal{S}$, we denote a set membership proof

$$\mu[C] \in \mathcal{S},$$

which reads μ encoded in commitment C is member of set $\mathcal{S}$.

B. Proof Techniques for the CG-encoding

Let us consider the basic principles underlying the proofs of the Camenisch-Gross (CG) encoding [20] in two examples on an Integer commitment

$$D = g^E h^r \bmod N.$$

Example 2 (Divisibility): Divisibility is the basis of an AND-Proof. To prove that a disclosed prime representative e_i is present in E, we prove that e_i divides the committed product E, we show that we know a secret μ' that completes the product:

$$e_i | E \quad \Leftrightarrow \quad e_i \cdot \mu' = E.$$

Expressed as proof of knowledge, we have:

$$PK\{(\mu', \rho) : \qquad D \equiv \pm (g^{e_i})^{\mu'} h^{\rho} \pmod N\}.$$

Example 3 (Coprimality): Coprimality is the basis of the NOT-Proof. We show that one or multiple prime representatives are not present in a credential, we show coprimality. To prove that two values E and F are coprime, i.e., $\mathsf{gcd}(E, F) = 1$, we prove there exist integers a and b such that Bézout's Identity equals 1:

$$\mathsf{gcd}(E, F) = 1 \quad \Leftrightarrow \quad aE + bF = 1.$$

Note that a and b for this equation do not exist, if $\mathsf{gcd}(E, F) > 1$.

$$\begin{aligned} PK\{(\mu, \rho, \alpha, \beta, \rho') : \quad & \pmod N \\ D \equiv \pm g^{\mu} h^{\rho} \quad \wedge \quad & g \equiv \pm D^{\alpha} (g^F)^{\beta} h^{\rho'}\}. \end{aligned}$$

a) AND-Proof: The CG-Encoding employs divisibility to show that one or more attributes, say e_i, e_j, and e_l, are encoded in an anonymous credential:

$$\begin{aligned} & PK\{(\varepsilon, \nu', \mu') : \\ & \quad Z \equiv \pm (R^{e_i e_j e_l})^{\mu'} A'^{\varepsilon} S^{\nu'} \pmod N \ \wedge \\ & \quad \mu' \in \pm\{0,1\}^{\ell_{\mathcal{M}} - 3\ell_t} \quad \wedge \quad \varepsilon \in [2^{\ell_e - 1} + 1, 2^{\ell_e} - 1]\}. \end{aligned}$$

b) NOT-Proof: We have seen that proving that a given e_j is not contained in the credential amounts to show that $e_j \nmid E$ is the case. The user can do so by showing that there exist two integers a and b such that $aE + be_j = 1$, where a and b do not exist if $e_j \mid E$. The values a and b can be computed efficiently with the extended Euclidian algorithm.

The CG-Encoding that achieves this is as follows: After having computed a and b, the prover computes an integer commitment $D = g^E h^r \bmod N$ with randomness r. The prover sends D to the verifier and runs the following protocol with him (where a and b are the secret denoted by α and β, respectively):

$$\begin{aligned} & PK\{(\varepsilon, \nu', \mu, \rho, \alpha, \beta, \rho') : \quad \pmod N \\ & \quad Z \equiv \pm R^{\mu} A'^{\varepsilon} S^{\nu'} \ \wedge \\ & \quad D \equiv \pm g^{\mu} h^{\rho} \ \wedge \ g \equiv \pm D^{\alpha} (g^{e_j})^{\beta} h^{\rho'} \ \wedge \\ & \quad \mu \in \pm\{0,1\}^{\ell_{\mathcal{M}}} \quad \wedge \quad \varepsilon \in [2^{\ell_e - 1} + 1, 2^{\ell_e} - 1]\}. \end{aligned}$$

c) OR-Proof: In addition, we can show that one of a list of attributes is contained in a credential (an OR-relation).

To prove that her credential contains one of the attributes values $\{e_1, ..., e_\ell\}$, a user can employ the following protocol. First, the user compute a commitment D to the attribute contained in here credential (in the same way as for the other protocols), send it to the verifier, and then runs with the verifier the following proof protocol.

Remark 4 (Ruling out ± 1): The proof employs a further group, i.e., one of prime order $\mathfrak{q}$ and two generators $\mathfrak{g}$ and $\mathfrak{h}$ of that group such that $\log_{\mathfrak{h}} \mathfrak{g}$ is unknown. Now, except the commitment D the attribute value in question, say e_j, as before, the user further computes the commitment $\mathfrak{D} = \mathfrak{g}^{e_j} \mathfrak{h}^r$, where r is a random element from $\mathbb{Z}_{\mathfrak{q}}$.
The full proof on the anonymous credential is formed as follows.

$$\begin{aligned} & PK\{(\varepsilon, \nu', \mu, \rho, \alpha, \beta, \delta, \rho, \rho', \varphi, \gamma, \psi, \xi, \sigma) : \quad \pmod N \\ & \quad Z \equiv \pm R^{\mu} A'^{\varepsilon} S^{\nu'} \ \wedge \quad D \equiv \pm g^{\alpha} h^{\rho} \ \wedge \\ & \quad g^{\prod_i^{\ell} e_i} = D^{\delta} h^{\rho'} \quad \wedge \quad 1 = D^{\beta} g^{\mu} h^{\rho'} \ \wedge \\ & \quad \mathfrak{D} = \mathfrak{g}^{\alpha} \mathfrak{h}^{\varphi} \quad \wedge \quad \mathfrak{g} = (\frac{\mathfrak{D}}{\mathfrak{g}})^{\gamma} \mathfrak{h}^{\psi} \quad \wedge \quad \mathfrak{g} = (\mathfrak{g}\mathfrak{D})^{\sigma} \mathfrak{h}^{\xi} \ \wedge \\ & \quad \mu \in \pm\{0,1\}^{\ell_{\mathcal{M}}} \quad \wedge \quad \varepsilon \in [2^{\ell_e - 1} + 1, 2^{\ell_e} - 1]\}. \end{aligned}$$