On Security Protocols for Desktop Sharing

Ulrich Kühn*
DZ BANK AG, Germany
Ulrich.Kuehn@dzbank.de
ukuehn@acm.org

Abstract: In this paper we examine security protocols employed in a number of tools for desktop sharing. These tools allow one user to see and interact with the desktop of another user, i.e. transmitting the contents of one computer's logical display to another place, including user interaction. In contrast to remote sessions, with desktop sharing, the access to the machine is shared, e.g. for interactive user support or for supporting administrators by experts for certain tasks or application programs. A number of these tools use an external communication server as a relay to sidestep problems when both the user and the support agent are behind firewalls.

In this paper we identify design flaws in the security protocols employed by a number of such tools, most notably a problem which allows the provider of the communication server to compromise the security of the communication. Further, we examine the certificates of security that some of these tools bear in the light of our findings. Additionally, we analyse the security requirements for a relayed communication protocol, which seems to be missing so far, and make high-level suggestions for an instantiation.

1 Introduction

With computers being prevalent on office worker's desktops, efficient support functionality is required. Here desktop sharing tools come into play, where a support agent can see and directly interact with what the user is being displayed on the screen. The advantage of using such tools is that the support agent can help a user who is located in a different location, as it is common for many of today's companies. A similar scenario for desktop sharing tools is remote administration with an outsourced administrator or specialised application expert. Here, the use of desktop sharing tools allows to implement a segregation-of-duty policy while maintaining support efficiency by avoiding travel.

Thus, the important difference between desktop sharing tools and remote sessions is that with desktop sharing, the session is actually shared, i.e. duplicated, so that at least two users can view the display contents and potentially interact with the system, while a remote session simply allows to log into a remote machine and interact with it. A kind of extension to desktop sharing is the web conferencing scenario, where more than one recipient participates. However, this group communication scenario is beyond the scope of this paper.

*Any opinions, findings, conclusions or recommendations expressed in this paper are those of the author and do not necessarily reflect the views of DZ BANK AG.

The communication protocols employed by desktop sharing or remote administration tools fall into two classes: First, protocols that need a direct connection between the accessed and the accessing computer which do not work with restrictive firewall settings, and second, communication protocols that employ relay servers as rendez-vous point and thus allow both communicating computers being behind firewalls that may block incoming traffic.

In this paper we are concerned with this second class of protocols that allow firewall piercing. We identify common design problems in these protocols and illustrate them by a case-study on actual products that contain these design problems. Further, as a number of these products come with a certificate of security, we examine the certificates in the light of our findings. Finally, we analyse the security requirements for such relayed communication protocols when it comes to desktop sharing. Then we suggest a high-level instantiation based on the identified security requirements.

Despite the considerable number of desktop sharing tools available we found that a treatment of the security of communication protocols which use a rendez-vous point for firewall piercing seems to be largely missing in the literature. Likewise, an analysis of the requirements regarding security of these protocols seems also to be missing in the literature.

2 Communication Models for Desktop Sharing: Direct vs. Relayed

The communication protocols employed by desktop sharing or remote administration tools can be distinguished into two classes, based on the way the communication is handled. In this paper we are concerned with the second class of relayed communication protocols.

Direct connection. The first class is comprised of protocols where the initiator needs to be able to directly open a network connection to the responder. Here the responder must be directly "visible" from the initiator's point of network view. Typically, the responder must not be behind a firewall that blocks incoming connections, at least for some ports. This scenario is rather standard.

Relayed communication. The second class of protocols consists of protocols which allow that both initiator and responder can be located behind restrictive firewalls. Here, the logical connection is made by resorting to some freely accessible communication intermediary. Both endpoints open *outgoing* connections, which are allowed in one way or another by most firewall setups. Here, the intermediary acts as a relay by passing the communication back and forth.

2.1 Entities Participating in Relayed Communication

Communication protocols that support relayed communication make use of intermediate communication servers, *relays*, which logically connect two connections with the communicating machines to form a logical connection. We denote the involved entities by

- The *initiator* is the entity which starts a communication. In the setup of interest here, the initiator connects to the relay, and lets it set up a connection. At this point in time the logical connection is prepared, but not fully existing.
- The *responder* is the other communicating entity. The responder also connects to the relay, but after the initiator.
- The *relay* actually implements the logical connection between initiator and responder by relaying data back and forth.

It should be noted that in fact there are three communication connections involved here: The connection between initiator and relay, the connection between responder and relay, and the logical connection between initiator and responder. These three connections need to be taken into account when designing and analysing security functions for relayed communication protocols.

3 Design Flaws in Desktop Sharing Protocols

Here we give a case study on some desktop sharing tools that employ relayed communication. For the security of the employed protocols it is of critical importance who owns and controls the relay. First, we analyse the security of the protocols when employing the vendor-provided relay, and second, when one of the communicating parties owns and controls the relay.

3.1 Common Flaws

Before we give details on the security protocols in our case study we want to highlight the problem areas that we identified:

Vendor-provided Relay as Man in the Middle. The protocols described and analysed below share the property of protecting the two connections between initiator and relay resp. responder and relay, but effectively not protecting the logical connection between initiator and responder. It should be noted that this man-in-the-middle issue is present despite the use of session tokens that are passed from the initiator to the responder both for identifying the correct session and for authenticating the responder. As a consequence, the security of the protocols does critically depend on who operates and controls the relay. For the first case considered here, i.e. the relay is controlled by neither of the communicating parties, this gives rise to the possibility of the relay acting as a man in the middle. Later we see that if one of the communicating parties owns and operates the relay, this problem is resolved, at the expense of losing the firewall-piercing property of the protocols for one of the communication legs.

Data Integrity. Another common issue, although not present in all the protocols analysed below, is the integrity of the transmitted data. It is well-known in the cryptographic community that encryption itself does *not* provide data integrity, and even a number of constructions with redundancy have been shown to fail [BN00, AB01].

3.2 Case Study on Protocols with Vendor-provided Relay

We want to highlight that here only a small selection of tools is described and analysed. In fact it seems difficult to obtain a complete list of such tools, just because the number of tools on the market. However, often no or only a very brief description of the security functions is given, e.g. the name and the key length of the symmetric cipher used for data encryption, but no further detail on key management, entity authentication etc.

Given that the tools we examine here are offered commercially, we did not have source code available. Instead we base our analysis on available descriptions and white papers, and, for some tools, personal communication with the manufacturer.

3.2.1 WebEx

The WebEx tool uses a relay that is implemented as a distribution network run by WebEx. In [Web05], section "Transport Layer Security", it is stated that the actual contents is encrypted using AES. However, it is unclear which entities have the key and if the contents is decrypted resp. re-encrypted by the relay. Further, the connections between initiator / responder and the relay can optionally be protected by SSL, apparently in addition the mentioned AES encryption. WebEx communicates using TCP port 1270, but allows to pass firewalls using ports 80 (HTTP) or 443 (HTTP over SSL).

While [Web05] does give some details about session authentication using passwords, no key management details are mentioned. As the session parameters are determined by the relay, it has to be assumed that all keys used for contents encryption are controlled by the relay as well.

Relay as Man in the Middle. WebEx states in [Web05] that it does not retain any session information. Thus, from the wording of the statement we assume that the relay does have access to contents in clear. Thus, effectively the logical connection between initiator and responder is not secured against the service provider WebEx itself. We did not find any statement indicating the contrary. Further, despite of the possibility of session passwords, no indication is given that the password is used as a shared secret between initiator and responder for precluding a man-in-the-middle attack.

Data Integrity. While SSL does have integrity protection built in, the termination of the secured channel at the relay precludes using it for end-to-end integrity protection. Further it is not clear if integrity protection is present if SSL is not used for content transmission.

3.2.2 Netviewer

The tool Netviewer comes as a stand-alone, installation-free program in two versions for initiator and responder. Its security functionality is described in a white-paper [Net08]. The security protocol for connection establishment works as follows: There are two pairs of asymmetric keys, one for the relay $(\mathrm{PK}_M, \mathrm{SK}_M)$, and one for the clients $(\mathrm{PK}_C, \mathrm{SK}_C)$, which are generated at installation time of the relay and are built into the client programs. The asymmetric encryption scheme involved here is an elliptic curve-based scheme with a claimed key length of 160 bits. However, the transmitted data offers space for at least 258 bits, and a deeper examination shows the use of $\mathrm{GF}(2^{255})$ as the underlying finite field[1]. The symmetric encryption employs Blowfish with 128 bit keys.

The connection setup protocol between initiator and relay is comprised of mutual authentication between relay and initiator based on the asymmetric key pairs. Then, the initiator sends a random mask value m to the server, encrypted under PK_M. The relay generates a symmetric session key k_I and sends $k_I \oplus m$, encrypted under PK_C, to the initiator. An analogous authenticated key exchange takes place between responder and relay. The resulting encrypted signalling channels are used to transmit another symmetric key k to the clients which they use for communication with each other. According to [Net08] the relay is comprised of a connection server, and a group of communication servers, with the key k never being transmitted to the communication servers. In order to cope with restrictive firewalls, the protocol can use port 80 (HTTP) or port 443 (HTTP over SSL). With the latter, an additional layer of protection between initiator / responder and relay is used.

Relay as Man in the Middle. The relay does generate the symmetric keys that are used to secure the communication, both on the signalling channels, and between initiator and responder. While [Net08] states that the key is not transmitted to the "communication server"-part of the relay, it is nevertheless in the position for a man-in-the-middle attack by retaining the key k, decrypting and modifying traffic between initiator and responder.

Data Integrity. While the white papers etc. do not provide any indication whether the integrity of the transmitted data is protected by cryptographic means, [Net09] indicates that HMAC-SHA1 [Nat02] is used. However, because of the relay generating the key and its position as man in the middle, this does not provide end-to-end integrity.

3.2.3 TeamViewer

The tool TeamViewer is comprised of stand-alone, installation-free programs for initiator and responder. Its security functionality is described in [Tea09]. It works roughly as follows:

[1] As the employed finite field has composite degree, the Gaudry-Hess-Smart attack and its extensions are relevant here. In fact, [MMT02] shows that the attack complexity is about 2^{52} operations, indicating a rather limited cryptographic security of the protocol. However, this problem is cured when tunnelling over SSL is used, see below. Interestingly, the documentation for the Common Criteria certificate (see Section 3.4) doesn't mention elliptic curve cryptography at all, and instead fully relies on SSL.

Both initiator and responder contain the relay's RSA public key PK_M, and subsequently generate their own RSA key pairs which they pass on to the relay for later use. On connection initiation the initiator requests the responder's public key from the relay. This exchange is encrypted with the respective public keys, and the response is signed by the relay. The initiator generates a symmetric key K, which it encrypts for the responder and sends it, signed with its private key, to the responder via the relay. The responder in turn obtains the initiator's public key from the relay, verifies the signature and decrypts the symmetric key K. After this protocol is run, both initiator and responder have a symmetric key which they use for further communication. The RSA keys are 1024 bits long, the symmetric encryption is done with AES-256.

Relay as Man in the Middle. We note that the relay acts as a trusted third party, certifying the initiator's and responder's public keys, which are, however, transmitted unauthenticated to the relay[2]. As a consequence the relay can act as a man in the middle, replacing the initiator's and responder's public key when answering the respective requests, and manipulate the transmission of the encrypted and signed symmetric key. Nevertheless, [Tea09] claims that not even the relay can decrypt the communication between initiator and responder. This claim is refuted by our analysis.

Data Integrity. Integrity protection of transmitted data is not mentioned in [Tea09]. Thus, it is not clear if integrity of the data is protected at all.

3.2.4 FastViewer

The FastViewer family of tools offers remote administration, desktop sharing and web conferencing. Here, we are here interested in the tool for desktop sharing and support. The FastViewer protocol distinguishes between two phases [Fas07], one for the rendez-vous of initiator and responder, and another one for the actual data communication. The rendezvous phase always uses the relay. The data communication phase can use direct connection or the relay, depending on network reachability.

Assuming that the data communication phase is using the relay, the second phase starts by a key exchange. While [Fas07] does state that a 256-bit AES key is exchanged between initiator and responder, it does *not* state how this is actually done. In [Fas09] some more protocol details were given: First, the initiator and responder both obtain a symmetric AES from the relay by generating and transmitting an RSA key to the relay, which sends back the AES key encrypted under the respective RSA key. These replies are signed by the relay and checked by the clients with a built-in verification key. Using the AES key, the resulting encrypted channel, another RSA key is transmitted from responder to initiator, which is in turn used to transfer the symmetric AES session key. Apparently, this AES session key is the one mentioned in [Fas07] with a length of 256 bits.

[2] It seems difficult to add here more than a self-signature of the respective client, if one does not want to require some form of registration or a client certificate. This would vastly complicate the use of the tool.

Relay as Man in the Middle. As the relay generates the first AES key, it can replace the second RSA key during transfer in order to obtain the session key. This is a classical man-in-the-middle attack. Given that apparently only a session number of 5 digits for authentication is exchanged using an out-of-bounds channel (e.g. telephone) between initiator and responder, it seems unlikely that the designers did consider the possibility of man-in-the-middle attacks.

Data Integrity. Integrity of transmitted data is not considered in [Fas07], neither was it mentioned in [Fas09]. Thus, we have to assume that no integrity protections are in place for the FastViewer tools.

3.3 Owning a Relay as a Partial Cure

Some of the tools examined above are offered with the option to deploy and use one's own relay, i.e. Netviewer, TeamViewer, and FastViewer.

In this case one of the communicating parties does have full control over the relay. This avoids having a third party, namely the operator of the relay, with control over the traffic. The fact that one of the communicating parties is controlling the relay heals the flaw identified in Section 3.2 where the operator of the relay as a third party can act as a man-in-the-middle due to the protocol design. However, at the same time the advantage of the relayed communication protocols is partly removed. Here, the party operating the relay must make sure that the relay can be reached by the other party from the outside world.

Nevertheless this operating model is well-suited for an enterprise scenario: The relay inside the enterprise network works as a central communication point. This allows to configure the firewall such that communication from outside is permitted to the relay, but there is no need to open the firewall to permit communication to all computers that potentially need remote support. Further, the initiator still fully controls if and when a desktop sharing session is possible.

3.4 Certification

Some of the tools are advertised with their security functionality being certified. Here we examine these certificates in the light of the findings presented above:

- The tool Netviewer (see Section 3.2.2) is advertised with a number of certificates: First, a certificate issued by Fiducia IT AG [FID06] on version 3.2 of Netviewer one2one presented as a "Security Certificate". However, it only certifies that the product is safe to use on a certain client setup for banking branches with no implication regarding security.

  Second, at the time of submission of this paper (March 2010), the tool was offered with a certificate issued by Fraunhofer Institut Sichere Informationstechnologie SIT

[Fra04] for version 2.0 of Netviewer one2one. Here, no evaluation criteria are given, and no special requirements are given for the user to fulfill. This certificate is listed as expired by the issuer [Fra]. However, the certificate does not identify the man-in-the-middle issue described above. This must be considered a flaw in the evaluation criteria or a failure in the evaluation process.[3]

Third, a common criteria certificate has been granted by Germany's Federal Office for Information Security (BSI) on version 5.1 of Netviewer one2one [Bun10] with an assurance level of EAL2 and a specific security target [Net09]. The security target lists the requirement that the relay is installed in a trustworthy environment. This fits well to our analysis in section 3.2.2. Nevertheless, the assurance level of EAL2 is rather low[4]. Nevertheless, the manufacturer's web site [Net10] contains the claim "experience shows that Netviewer is extremely secure". Interestingly, "experience" is cited, not the certificate.

- The tool FastViewer (see Section 3.2.4) is advertised with a certificate issued by TÜV Süd (see [Fas]). The certificate lists ISO/IEC25051:2009 for functionality and PPP13011:2004 for data security as certification criteria. The former is a public document containing requirements for software quality for commercial off-the-shelf software, but the latter not publicly available, not even upon request [Wol09]. Likewise, no evaluation report is available. In fact, the man-in-the-middle issue identified in Section 3.2.4 is not mentioned at all.

The certificates examined here, except the Common Criteria certificate for Netviewer, share the common problem that neither the certification criteria nor the certification report are not available for the customer. Thus, from a customer's perspective, these certificates do *not* provide enough information under which conditions these tools provide security against which attacks or adversaries. Consequently, these certificates are not helpful to understand the security the tools provide, and might lead to a false sense of security.

The exception here is the Common Criteria certificate with all the necessary information available. However, the details of security target and the assurance given by the EAL2 certification have to be evaluated carefully if they are suitable for the intended usage scenario.

4 Designing a Relayed Security Protocol for Desktop Sharing

Here we state the basic security requirements for relayed communication protocols and give a high-level description of building blocks for instantiation.

[3] Another certificate by the same issuer on another desktop sharing tool (not included in the case study above for lack of details on the security protocol) state as condition "provided the [...] server component is located in a protected and trustworthy environment" [Fra06]. The inclusion of this condition hints to an update in the (unpublished) evaluation criteria.

[4] [Bun10] explains that EAL2 is "applicable [...] where [...] users require a low to moderate level of independently assured security in the absence of ready availability of the complete development record. Such a situation may arise when securing legacy systems [...]"

4.1 Requirement Analysis and Protocol Flow Proposal

As indicated in Section 3.1 with relayed communication there are not only two but effectively three connections involved which must be secured: First, Initiator ↔ Relay, second, Responder ↔ Relay, and last, Initiator ↔ Responder via the relay. In fact there is another – out-of-band – channel between initiator and responder over which the responder obtains information on how to make the relay connect the responder's to the initiator's session. In a remote support scenario this out-of-band channel would be provided by telephone.

The general setup protocol would flow as follows, where the first seven steps are in place to connect the initiator and responder via the relay, and the last step addresses the man-in-the-middle issue by securing the logical connection:

1. The initiator connects to the relay, which sets up an (unconnected) session s.
2. The relay provides some session token t_s to the initiator.
3. The initiator passes t_s to the responder using an out-of-band channel.
4. The responder connects to the relay which sets up an (unconnected) session s'.
5. The responder forwards t_s to the relay.
6. The relay uses t_s to identify the session s and logically connects sessions s and s'.
7. The relay signals both the initiator and responder that the sessions has been successfully connected.
8. Initiator and responder use the logical channel to set up the session between them.

Note that there is no need for a session token for s' if the responder identifies itself as a responder. In this protocol framework four steps, namely steps 2, 5, 6, and 8, give rise to security requirements:

R1. As t_s is an authenticator for connecting to the session s it must be delivered in a confidential way, first to the initiator in step 2, then to the responder using the out-of-band channel, and lastly to the relay in step 5.

R2. t_s must be constructed such that the relay can reliably determine its authenticity, its freshness and its connection to the session s in step 6.

R3. For transmitting out of band in step 3, the token t_s must be short and easy to communicate by humans.

R4. The logical connection between initiator and responder must be secure, providing both confidentiality and integrity / authenticity of transmitted data. To preclude man-in-the-middle attacks the keys and authentication information must not be solely based on information exchanged with or with the help of the relay.

R5. Depending on the business model of the entity providing the relay the initiator resp. the responder optionally must first successfully authenticate to the relay in step 1 resp. 4 to show that he or she is entitled to used the relay's services.

4.2 High-level Description of Protocol Instantiation

Here we provide a high-level sketch of how the protocol framework could be instantiated and how the security requirements could be fulfilled. For a secure instantiation of the framework protocol there are three building-blocks to be specified:

Securing the connections between initiator and relay resp. responder and relay. We propose using SSL/TLS with a server certificate for the relay. This provides secrecy and integrity of the transmitted data as well as authenticity of the relay for this control connection. As it is a point-to-point connection, the SSL/TLS standard is appropriate. Optionally, the initiator could authenticate against the relay by using user-ID / password or a certificate. This way a business model with pay-per-use could be set up for the relay. This choice addresses security requirements R1 and R5.

Designing a secure scheme for the session token. Given the requirements for the session token, we propose the following construction, based on the assumption that about 40 bits of information can be passed by human communication when properly encoded.

When setting up a session s with an initiator, the relay generates state information $I_s = (N, T, K)$ consisting of a session number N, a time T of establishing I_s, and a MAC key K. K is generated from a cryptographically secure random bit generator. The session number N is 20 bits long, allowing 2^{20} parallel open sessions, and is either generated randomly (excluding collisions with existing session numbers) or from a counter. The session number can theoretically be reused after the full connection is made.

The session token $t_s = (N, \tau)$ is computed by $\tau = [(\mathrm{MAC}_K(N, T)]_{20}$ where $[.]_x$ means truncation to x bits, $I_s = (N, T, K)$, and MAC is a secure MAC scheme, such as CBC-MAC [ISO99] based on AES or HMAC [Nat02]. We propose to use alpha-numerical characters for encoding t_s, i.e. 5 bits per character, yielding a string of 8 characters. Such a string can be easily transmitted by telephone.

Verifying a session token $t_s = (N, \tau)$ during step 5 of the protocol framework requires the relay to find session information $I' = (N', T', K')$ such that $N' = N$ and verifying that $\tau = [(\mathrm{MAC}_{K'}(N', T')]_{20}$. Further, it checks that T is not older than a short time-out. Here, the probability of success of an adversary is roughly $1/2^{-20}$ if it successfully identifies an open session. In our view this provides sufficient protection, given that the actual authentication between responder and initiator is done in step 8. Actually, an adversary passing the token verification can be seen as a residual risk of the adversary using the relay. Possibly the authenticator τ could be truncated even more. This construction addresses security requirements R2 and R3.

Securing the logical connection between initiator and responder. Any authenticated key exchange method could be used here. However, we propose to employ the out-of-band channel between initiator and responder also for this task, making sure that the entity whose machine needs to be accessed receives the authentication information:

First a key exchange, e.g. Diffie-Hellman, would run between the two entities. Second, an authenticator string would be computed using the computed shared secret from the entities public keys or all messages of the key exchange. The authenticator would then be transmitted by the accessing entity to the accessed entity using the out-of-band channel. Finally, access would be granted after successfull verification by the accessed entity. This provides a corroboration to the accessed entity that the right entity is on the other end of the connection, thereby addressing security requirement R4.

5 Conclusion

In this paper we have first analysed the security of desktop sharing and remote support tools that employ relayed communication. Such communication patterns allows parties to communicate even if one or both of them are behind firewalls.

It turned out that with this kind of indirect communication new security issues arise that are not addressed by usual secure communication protocols like SSL/TLS. We have examined a selection of tools for desktop sharing and remote support, where we identified common security issues. Furthermore, we found that in an enterprise environment, placing and controlling the relay inside the enterprise removes this fundamental issue and, additionally, allows a better-controlled configuration of firewalls.

In the second part of this paper we did an analysis of the security requirements for relayed communication and proposed a framework and possible high-level instantiations of the security building blocks.

To summarise, our analysis shows that designing a secure relayed communication protocol is not a trivial task and that trusting the relay without securing the logical connection between the communicating entities can allow the relay to act as a man in the middle, thus threatening the privacy and integrity of the communication.

Future research could be directed towards the related scenario of web conferencing, where more than one responder is present. Here relayed communication has to be combined with secure group communication protocols.

References

[AB01] Jee Hea An and Mihir Bellare. Does Encryption with Redundancy Provide Authenticity? In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT '2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 509–524, Innsbruck, Austria, 2001. Springer-Verlag, Berlin Germany.

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In T. Okamoto, editor, *Advances in Cryptology – ASIACRYPT '2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545, Kyoto, Japan, 2000. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany.

[Bun10] Bundesamt für Sicherheit in der Informationstechnik (BSI). Zertifizierungsreport BSI-DSZ-CC-0524-2010, March 2010. https://www.bsi.bund.de/cae/servlet/contentblob/950786/publicationFile/61801/0524a_pdf.pdf.

[Fas] FastViewer. TÜV Süd Certificate (2009). http://www.fastviewer.com/awards.html.

[Fas07] FastViewer. Erklärung zum Verschlüsselungsverfahren und zur Datensicherheit beim Einsatz von FastViewer, October 2007. http://www.additive-net.de/ftp/win32/software/fastviewer/ADD_ES_FSV_Verbindungsaufbau_und_Sicherheit.pdf.

[Fas09] FastViewer. Personal Communication, 2009.

[FID06] FIDUCIA IT AG. Bestätigung Sicherheitstechnische Prüfung von Fremdsoftware – Netviewer one2one und one2meet, Juni 2006. http://www.genodata.de/support/download/zertifikate/Fiducia%20IT%20AG_Zertifikat%20one2one%20one2meet.pdf.

[Fra] Fraunhofer Institut Sichere Informationstechnologie. List of Certificates. http://testlab.sit.fraunhofer.de/content/output/certificates.php.

[Fra04] Ergebnis der Prüfung der Netviewer Software durch das Fraunhofer Institut für Sichere Telekooperation, April 2004. http://testlab.sit.fraunhofer.de/downloads/certificates/netviewer%20200404.pdf.

[Fra06] Certificate on pcvisit 4 certified security version 4.1.0.1476, July 2006. http://testlab.sit.fraunhofer.de/downloads/certificates/Urkunde%20pcvisit%20en%2006-104701.pdf.

[ISO99] ISO/IEC. IS 9797-1: Information Technology – Security techniques – Message authentication codes (MACs) – Part 1: Mechanisms using a block cipher, 1999.

[MMT02] Markus Maurer, Alfred Menezes, and Edlyn Teske. Analysis of the GHS Weil Descent Attack on the ECDLP over Characteristic Two Finite Fields of Composite Degree. *LMS J. Comput. Math*, 5:127–174, 2002.

[Nat02] National Institute of Standards and Technology (NIST). The Keyed-Hash Message Authentication Code. Federal Information Processing Standards Publication (FIPS PUB) 198, March 2002.

[Net08] Netviewer. White Paper Security: Netviewer Support, Version 1.6, September 2008. http://www.netviewer.de/fileadmin/PDF/whitepaper/Netviewer_Whitepaper_security_Netviewer_Support_EN.pdf.

[Net09] Netviewer. Sicherheitsvorgaben (Security Target) zu Netviewer one2one Version 5.1. https://www.bsi.bund.de/cae/servlet/contentblob/950784/publicationFile/61802/0524b_pdf.pdf, September 2009. BSI-Zertifizierungs-ID BSI-DSZ-CC-0524.

[Net10] Netviewer. Web site, May 2010. http://www.netviewer.com.

[Tea09] Teamviewer GmbH. TeamViewer Security Information, May 2009.

[Web05] WebEx Communications Inc. WebEx Security Overview, February 2005. http://www.webex.com/pdf/wp_security_overview.pdf.

[Wol09] Wolf-Rüdiger Heidemann (TÜV Süd). Personal Communication, July 2009.