User-aided data authentication

Sven Laur*

Department of Computer Science,
University of Tartu,
Liivi 2, 50409 Tartu, Estonia
E-mail: swen@math.ut.ee
*Corresponding author

Sylvain Pasini

The Security and Cryptography Laboratory (LASEC),
Swiss Federal Institute of Technology (EPFL),
CH-1015 Lausanne, Switzerland
E-mail: sylvain.pasini@epfl.ch

Abstract: All classical authentication protocols are based on pre-shared authentic information such as long-term secret keys or a public key infrastructure. However, there are many practical settings, where participants can additionally employ authentic Out-Of-Band (OOB) communication, e.g., manual message transfer. In this paper, we study the corresponding user-aided message authentication and key agreement protocols. In particular, we give a unified treatment of many previous results and outline common design principles. We also show that certain properties of user-aided protocols simplify the security analysis in complex environments compared to the standard authentication protocols.

Keywords: message authentication; key agreement; group; universal composability.

Reference to this paper should be made as follows: Laur, S. and Pasini, S. (2009) 'User-aided data authentication', *Int. J. Security and Networks*, Vol. 4, Nos. 1/2, pp.69–86.

Biographical notes: Sven Laur received his Doctoral Degree in the Helsinki University of Technology. He received his Master's Degree in Computer Science from the University of Tartu in 2003.

Sylvain Pasini is currently doing a PhD thesis at the Swiss Federal Institute of Technology (EPFL) in the Security and Cryptography Laboratory (LASEC). He received his Master's Degree in Communication Systems from the Swiss Federal Institute of Technology in 2005 after receiving a diploma from the Engineering School of Geneva (EIG) in 2001.

1 Introduction

One of the most important tasks in cryptography is to establish a secure communication over an insecure channel between two or more participants. This can be trivially done when all users are sharing a secret key. Indeed, they can use any secure block or stream cipher to protect the data communications. In reality, the challenging problem is how to establish a common secret key.

With no assumptions, participants must exchange the secret key directly. For that, at least once before the communication they need to use a secure extra channel that ensures confidentiality, authenticity and integrity. Such a channel is very expensive in practice because participants must be physically close while the keys are exchanged.

On the same idea, assume that participants share a low-entropy secret, also known as a password. In this setting, participants can run a password-based key agreement protocol. Then, at the end of the execution, they obtain a high-entropy shared secret key. The problem is the same as before: they must use a secure extra channel. The only gain compared to the first solution is that the secure channel is used to exchange a smaller amount of data.

The use of public-key primitives can relax the confidentiality assumption on the extra channel. For instance, the Diffie-Hellman key exchange protocol (Diffie and Hellman, 1976) is a secure way to establish a secret key under the standard complexity-theoretic assumptions. However, the Diffie-Hellman protocol is insecure against man-in-the-middle attacks. For that reason, we must authenticate the protocol transcript. Another example is the use of any secure public-key cryptosystem such as RSA (Rivest et al., 1978) or ElGamal (ElGamal, 1985). In this case, we must assure that a public key is transferred to all

participants in an authenticated way. In a nutshell, setting up a secure communication can be reduced to the problem of authenticating messages. Indeed, as long as we are able to authenticate data, we can establish a shared secret key and thus we can also protect communication over insecure channels.

We emphasise that transcripts of common key agreement protocols are usually several thousands bits long and thus message authentication is a non-trivial task. Of course, we can use message authentication codes but this requires a shared secret key that we are only trying to establish. Alternatively, we can use digital signatures for that purpose but they also require authentic transfer of public keys. Consequently, the use of digital signatures can only reduce the amount of essential authentic communication. In particular, we can envisage the use of a Public-Key Infrastructure (PKI) that will deliver certificates for public keys used for signing. Here we clearly note two main disadvantages: firstly, this solution requires a huge infrastructure which is expensive, and secondly, it requires us to trust external parties, e.g., certificate authorities.

In other words, we cannot authenticate messages without relying on authenticated channels. Hence, many practical communication protocols such as SSH, PGP, Bluetooth and WUSB use extra channels which achieve at least authentication. Indeed, in SSH and PGP, the public keys are authenticated by the help of the user who will check the fingerprints. When a user receives a PGP public key, he or she computes its fingerprint and then calls the claimed owner of the public key. They check if both fingerprints are equal. If they are then the public key is authenticated, otherwise the key must have been altered.

More formally, use-aided protocol is a protocol that directly uses Out-Of-Band (OOB) messages. Namely, we assume that most messages are transmitted over insecure channels, referred as the *in-band communication*, while some authentic data is transmitted over an extra channel, referred as the *out-of-band communication*. Normally, the OOB channel is established by a human operator. For instance, a user can establish OOB communication channel by doing relatively simple tasks like copying a string from one device to another or spelling a string by phone. Indeed, such tasks create authenticated channels in practice, since no adversary controlling the network can forge these OOB messages. On the other hand, these protocols require the help of the user and thus they should request only small tasks in order to stay user-friendly.

In consequence, protocol designers should use the minimum amount of OOB data as they can for the desired security level. This line of research was initiated by Balfanz et al. (2002) who were the first to formalise the fingerprint protocol discussed above. However, first non-trivial results were obtained by Gehrmann et al. (2004) who showed how to construct user-aided authentication protocols that preserve reasonable security levels even for short OOB messages consisting of 4–6 decimal digits. The latter made user-aided data authentication practical for securing short-range wireless communication such as Bluetooth and WiFi networks. The SAS protocol proposed by Vaudenay (2005) was the second important discovery. The protocol was the first to achieve optimal security level and thus halved the length of required OOB messages. Vaudenay also introduced the concept of user-aided protocols to wider cryptographic audience under the name of SAS-based cryptography where SAS stands for Short Authenticated Strings.

Our contribution

Our main contribution in this paper is a systematised overview of various user-aided message authentication protocols and their applications. In particular, we show how our earlier results (Pasini and Vaudenay, 2006b; Laur and Nyberg, 2006; Laur and Pasini, 2008) fit into the general framework of user-aided data authentication.

In Section 2, we describe cryptographic primitives that play essential role in user-aided message authentication protocols. Namely, keyed hash functions with information theoretical properties are needed to handle short out-band-messages and commitment schemes are needed to achieve optimal deception level. In Section 3, we formalise the notion of user-aided data authentication and show how to express various functional requirements in the stand-alone security model. The latter is an important methodical advance, since one can pose many complex design requirements on message authentication protocols.

Section 4 provides a systematised overview of two-party message authentication protocols including our earlier results (Pasini and Vaudenay, 2006b; Laur and Nyberg, 2006). As an important theoretical result, we show that all state of the art user-aided authentication protocols share the same internal structure. Namely, they use in-band communication and OOB messages to mimic the behaviour of classical authentication protocols with pre-shared secret keys. The shared internal structure also explains why all protocols with optimal deception bound rely on the same security premises and why non-malleability of used commitments is so important. In Section 5, we show how to extend these ideas to group settings. In particular, we describe the underlying structure of our SAS-GKA protocol (Laur and Pasini, 2008).

As a second important theoretical result, we prove that stand-alone security guarantees are preserved in complex settings as long as a simple set of usage restriction are satisfied. The latter significantly simplifies the security analysis of user-aided message authentication protocols. These issues are thoroughly discussed in Section 6, where we clarify the relations between various security models.

More precisely, we show that all message authentication protocols are universally composable as soon as they are secure in the stand-alone model. However, the latter is not sufficient for security when several protocols share the same trusted setup phase. As a result, one must use the Bellare-Rogaway model in order to analyse security of practical applications that reuse the same secret key many times. Notably, user-aided message authentication

protocols are different in that respect, since they do not rely on common secrets and preserve stand-alone security guarantees even in the Bellare-Rogaway model.

In Section 7, we discuss how to employ user-aided message authentication protocols to protect key agreement protocols against active attacks. We also describe some additional measures that decrease the amount of required user-interaction in dynamical settings.

Finally, this paper ends with some concluding remarks and some open questions in Section 8. In particular, we comment the tension between theoretical results and practical implementations of various user-aided authentication and key agreement protocols. Namely, all security proofs in this paper are given by using the weakest abstract properties, however, one often substitutes these primitives with heuristic constructions in practice. Hence, understanding the corresponding risks and limitations is important.

2 Cryptographic preliminaries

All results in this paper are stated in terms of exact security. That is, security properties are always specified by a game or a game pair between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. For a single game $\mathcal{G}$, the advantage is defined as $\mathrm{Adv}(\mathcal{A}) = \Pr[\mathcal{G}^{\mathcal{A}} = 1]$. For a game pair $\mathcal{G}_0, \mathcal{G}_1$, the advantage is defined as $\mathrm{Adv}(\mathcal{A}) = |\Pr[\mathcal{G}_0^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_1^{\mathcal{A}} = 1]|$. Typically, one requires that for all t-time adversaries $\mathcal{A}$, the corresponding advantage is bounded: $\mathrm{Adv}(\mathcal{A}) \leq \varepsilon$.

Although exact quantification of security properties is our main goal, we use asymptotic estimates to hide irrelevant technical details. Note that these bounds are given in the setting, where the cryptographic construction is fixed and only the adversarial computational power t varies. Of course, these results can be translated back to the non-uniform polynomial security model by considering asymptotics with respect to the security parameter.

2.1 Keyed hash functions

A keyed hash function $h : \mathcal{M} \times \mathcal{R} \to \mathcal{T}$ takes two arguments: a message $m \in \mathcal{M}$ and a key $r \in \mathcal{R}$, and outputs a digest $t \in \mathcal{T}$. Keyed hash functions are commonly used as building blocks in message authentication protocols. For example, two participants who share a secret key $r \in_{\mathrm{u}} \mathcal{R}$ can add a digest t to the message to protect it from tampering. Now a potential adversary can carry out two types of attacks. First, the adversary might try to impersonate a key holder by creating a valid message tag pair $(\hat{m}, \hat{t})$ without no additional information. Secondly, the adversary might try to substitute a message m by altering the corresponding pair (m, t). The security against impersonation and substitution attacks depends on the regularity and the universality of the hash function. A hash function h is ε_{r}-*almost regular* if for any $m \in \mathcal{M}$ and $t \in \mathcal{T}$:

$$\Pr[r \in_{\mathrm{u}} \mathcal{R} : h(m, r) = t] \leq \varepsilon_{\mathrm{r}}.$$

A hash function h is ε_{u}-*almost universal*, if for any two distinct inputs $m_0 \neq m_1$:

$$\Pr[r \in_{\mathrm{u}} \mathcal{R} : h(m_0, r) = h(m_1, r)] \leq \varepsilon_{\mathrm{u}}$$

and ε_{u}-*almost XOR-universal*, if for any two distinct inputs $m_0 \neq m_1$ and a difference $\Delta t \in \mathcal{T}$:

$$\Pr[r \in_{\mathrm{u}} \mathcal{R} : h(m_0, r) \oplus h(m_1, r) = \Delta t] \leq \varepsilon_{\mathrm{u}}.$$

These notions can be extended to hash functions with many sub-keys, i.e., for $h : \mathcal{M} \times \mathcal{R}_1 \times \cdots \times \mathcal{R}_n \to \mathcal{T}$. A function h is ε_{u}-*almost universal w.r.t. the sub-key* r_i if for any input pair $m_0 \neq m_1$ and sub-keys $r_j, \hat{r}_j \in \mathcal{R}_j$:

$$\Pr[r_i \in_{\mathrm{u}} \mathcal{R}_i : h(m_0, \boldsymbol{r}) = h(m_1, \hat{\boldsymbol{r}})] \leq \varepsilon_{\mathrm{u}},$$

where $\boldsymbol{r}$ denotes a vector $(r_1, \ldots, r_{i-1}, r_i, r_{i+1}, \ldots, r_n)$ and $\hat{\boldsymbol{r}}$ denotes a vector $(\hat{r}_1, \ldots, \hat{r}_{i-1}, r_i, \hat{r}_{i+1}, \ldots, \hat{r}_n)$. A hash function h is ε_{u}-*almost universal w.r.t. the sub-key pairs*, if for any input pair $m_0 \neq m_1$, indices i, j and $r_j, \hat{r}_j \in \mathcal{R}_j$:

$$\Pr[r_* \in_{\mathrm{u}} \mathcal{R} : h(x_0, \boldsymbol{r}) = h(x_1, \hat{\boldsymbol{r}})] \leq \varepsilon_{\mathrm{u}},$$

where $\boldsymbol{r}$ denotes a vector $(r_1, \ldots, r_{i-1}, r_*, r_{i+1}, \ldots, r_n)$ and $\hat{\boldsymbol{r}}$ denotes a vector $(\hat{r}_1, \ldots, \hat{r}_{j-1}, r_*, \hat{r}_{j+1}, \ldots, \hat{r}_n)$ and the equality $i = j$ is allowed. Finally, a hash function is ε_{r}-*almost regular w.r.t. to the sub-key* r_i, if for any input m, sub-keys $\hat{r}_j \in \mathcal{R}_j$ and a target digest $t \in \mathcal{T}$:

$$\Pr[r_i \in_{\mathrm{u}} \mathcal{R}_i : h(m, \hat{r}_1, \ldots, r_i, \ldots, \hat{r}_n) = t] \leq \varepsilon_{\mathrm{r}}.$$

All protocols presented in this paper use hash functions that are both ε_{r}-almost regular and ε_{u}-almost universal. It is straightforward to prove that $\varepsilon_{\mathrm{r}}, \varepsilon_{\mathrm{r}} \leq 2^{-\ell}$ if the digest t can be at most ℓ bits long. However, it is also possible to find hash functions that achieve optimality $\varepsilon_{\mathrm{r}} = \varepsilon_{\mathrm{r}} = 2^{-\ell}$ or are almost optimal. See the papers (Laur and Nyberg, 2006; Pasini and Vaudenay, 2006b; Laur and Pasini, 2008) for further discussion.

2.2 Commitment schemes

A commitment scheme is another important building block in many user-aided message authentication protocols. A *commitment scheme* $\mathcal{C}om$ is specified by a triple of algorithms $(\mathsf{setup}, \mathsf{com}, \mathsf{open})$. The setup algorithm setup generates public parameters pk for the commitment scheme. The commitment algorithm $\mathsf{com}_{\mathsf{pk}} : \mathcal{M} \to \mathcal{C} \times \mathcal{D}$ maps messages $m \in \mathcal{M}$ into a commitment string $c \in \mathcal{C}$ of fixed length and a decommitment value $d \in \mathcal{D}$. Usually the decommitment value is a pair $d = (m, r)$ where r is the randomness used to compute c. A commitment scheme is functional if for all $(c, d) \leftarrow \mathsf{com}_{\mathsf{pk}}(m)$ the equality $\mathsf{open}_{\mathsf{pk}}(c, d) = m$ holds. Incorrect decommitment values should yield a special abort value $\perp$.

Three most commonly used cryptographic properties of commitment schemes are hiding, binding and non-malleability. Non-malleability is the strongest property, since binding and hiding properties directly follow from non-malleability and not vice versa. Many notions of non-malleable commitments have been proposed in cryptographic literature (Dolev et al., 1991; Crescenzo

et al., 1998; Fischlin and Fischlin, 2000; Damgård and Groth, 2003; Laur and Nyberg, 2006). All these definitions try to capture requirements that are necessary to defeat man-in-the-middle attacks. In this work, we adopt the modernised version of non-malleability w.r.t. opening. The corresponding definition (Laur and Nyberg, 2006) mimics the framework of non-malleable encryption (Bellare and Sahai, 1999) and leads to more natural security proofs compared to the simulation based definitions (Crescenzo et al., 1998; Damgård and Groth, 2003).

Non-malleability and security against Chosen Ciphertext Attacks (CCA) are known to be tightly coupled. In fact, these notions coincide if the adversary is allowed to make decryption queries throughout the entire attack (Bellare et al., 1998) and thus usage of decryption oracles can simplify many proofs without significantly increasing the security requirements. Unfortunately, a similar technique is not applicable to commitment schemes as there can be several different valid decommitment values d_i for a single commitment c. Thus, we must use explicit definitions of hiding, binding and non-malleability properties in the following security proofs.

A commitment scheme $\mathcal{C}om$ is $(t, \varepsilon_{\mathrm{h}})$-*hiding* if any t-time adversary $\mathcal{A}$ succeeds in the hiding game with probability at most ε_{h}, i.e., $\mathrm{Adv}^{\mathsf{hid}}_{\mathcal{C}om}(\mathcal{A}) \leq \varepsilon_{\mathrm{h}}$ where

$$\mathrm{Adv}^{\mathsf{hid}}_{\mathcal{C}om}(\mathcal{A}) = 2 \cdot \left| \Pr \left[\begin{array}{l} \mathsf{pk} \leftarrow \mathsf{setup}, s \in_{\mathrm{u}} \{0,1\}, \\ (x_0, x_1, \sigma) \leftarrow \mathcal{A}(\mathsf{pk}), \\ (c_s, d_s) \leftarrow \mathsf{com}_{\mathsf{pk}}(x_s) : \\ \quad \mathcal{A}(\sigma, c_s) = s \end{array} \right] - \frac{1}{2} \right|.$$

A commitment scheme $\mathcal{C}om$ is $(t, \varepsilon_{\mathrm{b}})$-*binding* if any t-time adversary $\mathcal{A}$ succeeds in the binding game with probability at most ε_{b}, i.e., $\mathrm{Adv}^{\mathsf{bind}}_{\mathcal{C}om}(\mathcal{A}) \leq \varepsilon_{\mathrm{b}}$ where

$$\mathrm{Adv}^{\mathsf{bind}}_{\mathcal{C}om}(\mathcal{A}) = \Pr \left[\begin{array}{l} \mathsf{pk} \leftarrow \mathsf{setup}, (c, d_0, d_1) \leftarrow \mathcal{A}(\mathsf{pk}) : \\ \mathsf{open}_{\mathsf{pk}}(c, d_0) \neq \mathsf{open}_{\mathsf{pk}}(c, d_1), \\ \mathsf{open}_{\mathsf{pk}}(c, d_i) \neq \perp \text{ for } i \in \{0,1\} \end{array} \right].$$

The non-malleability property is defined by complicated games, and thus we use an illustrative pictorial style to specify these games, see Figure 1. Intuitively, the goal is: given a valid commitment c, it is infeasible to generate related commitments $\hat{c}_1, \ldots, \hat{c}_n$ that can be successfully opened after seeing a decommitment value d. More formally, the adversary $\mathcal{A}$ consists of two parts: $\mathcal{A}_1$ corresponds to the *active* part of the adversary that tries to create and afterwards open commitments related to c while $\mathcal{A}_2$ captures a desired *target relation*. Note that $\mathcal{A}_1$ is a stateful algorithm and can pass information from one stage to the other but no information can be passed from $\mathcal{A}_1$ to $\mathcal{A}_2$ except σ. By convention, a game is ended with the output $\perp$ if any operation leads to $\perp$.

Figure 1 should be read as follows. In the game $\mathcal{G}^{\mathsf{nm}}_0$, a challenger $\mathcal{C}$ first generates the public parameters pk. Given pk, the adversary outputs a message generator MGen. Next, the challenger selects $x_0 \leftarrow \mathsf{MGen}$ and computes (c, d). Given c, the adversary outputs some commitment values $\hat{c}_i$ and an advice σ for $\mathcal{A}_2$ and then, given d he generates some decommitment values $\hat{d}_i$. Finally, the challenger opens all commitments $\hat{y}_i \leftarrow \mathsf{open}_{\mathsf{pk}}(\hat{c}_i, \hat{d}_i)$ and tests whether $\mathcal{A}_1$ won or not by computing $\mathcal{A}_2(\sigma, x_0, \hat{y}_1, \ldots, \hat{y}_n)$. The condition $\hat{c}_j \neq c$ eliminates trivial attacks. The game $\mathcal{G}^{\mathsf{nm}}_1$ is almost the same, except the challenger tests a relation $\mathcal{A}_2(\sigma, x_1, \hat{y}_1, \ldots, \hat{y}_n)$ instead, where $x_1 \leftarrow \mathsf{MGen}$ is chosen independently from the rest of the game. A commitment scheme is $(t, \varepsilon_{\mathrm{nm}})$-*non-malleable* w.r.t. to opening if for any t-time adversary $\mathcal{A}$ the advantage

$$\mathrm{Adv}^{\mathsf{nm}}_{\mathcal{C}om}(\mathcal{A}) = |\Pr[\mathcal{G}^{\mathsf{nm}}_0 = 1] - \Pr[\mathcal{G}^{\mathsf{nm}}_1 = 1]| \leq \varepsilon_{\mathrm{nm}}.$$

Note that $\mathcal{A}_2$ can be any computable relation that is completely fixed after seeing c. For instance, we can define $\mathcal{A}_2(\sigma, x, y) = [x \stackrel{?}{=} y]$. Hence, it must be infeasible to construct a commitment $\hat{c}$ that can be opened later to the same value as the challenge commitment c.

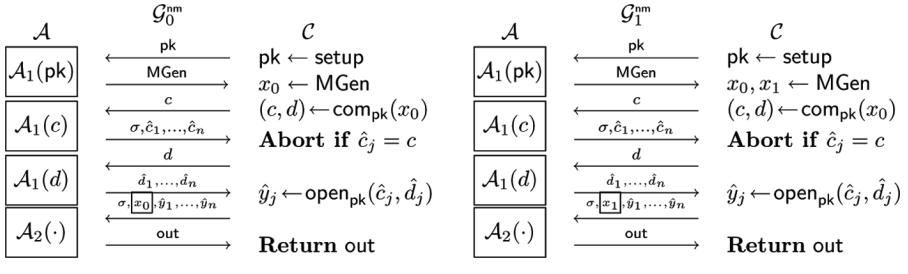
Non-malleable commitments schemes can be easily constructed based on simulation-sound trapdoor commitments from MacKenzie and Yang (2004) as detailed by Vaudenay (2005). They can also be built using a CCA2 secure encryption scheme, or by using a hash function as detailed by Laur and Nyberg (2006).

2.3 Common reference string model

We emphasise that all security definitions for commitment schemes explicitly assume that system wide initial parameters pk are generated by a trusted third party. Such a setting is known as Common Reference String (CRS) model. Although this assumption is not essential, most of the communication and computation efficient commitment schemes are specified for the CRS model.

The CRS model is not so restrictive as it seems at first glance. All communication standards provide system wide public parameters such as specifications of hash functions or a bit length of public keys. Therefore, one should make a trade-off between computational efficiency and reusability

Figure 1 Non-malleability games $\mathcal{G}^{\mathsf{nm}}_0$ and $\mathcal{G}^{\mathsf{nm}}_1$



and size of system-wide parameters pk. Moreover, there are theoretic constructions that allow generation of a common reference string in the standard model.

3 Message authentication protocols

Authentication protocols are used for two main purposes. We can either protect authenticity of communication between two or more participants or alternatively verify that a person or a device is who he, she or it claims to be. These security goals are somewhat orthogonal, since message authentication protocols only assure that messages are not tampered during the transmission. Although the latter also identifies the corresponding physical sender devices, it does not automatically guarantee validity of claimed legal identities. Hence, entity authentication is often done separately after all communication links are secured. In this paper, we investigate only message authentication.

As usual, we assume that the actions of all participants including potential adversaries depend only on the received messages and their relative ordering. This assumption is often justified even if a practical instantiation of a protocol depends on explicit timings. In fact, it is straightforward to prove that security guarantees obtained in this simplified model are valid for all practical settings, where exact timings do not depend on the states of private variables.

3.1 Communication model

It is often prohibitively expensive to establish necessary physical infrastructure that guarantees integrity of received messages. Authenticity concerns are particularly justified in case of wireless communication, since anybody with right equipment can eavesdrop, inject messages and cause communication failures. Thus, we have to assume that participants exchange messages over a communication network that is controlled by a malicious adversary. However, the latter does not exclude possibility of truly authentic message transmission, since participants may use alternative ways to communicate. For instance, in many small-range wireless networks a human operator can authentically transfer short messages from one device to another. If entities are further apart, we can transfer such messages over the phone provided that participants can recognise each other by voice and behaviour.

As usual, we consider a model where communication is asynchronous. Parties can use *in-band* and *out-band* communication channels. In-band communication is insecure and routed via an active adversary $\mathcal{A}$ who can drop, delay, modify and insert messages. Additionally, parties can send Short Authenticated Strings (SAS) aka OOB messages. OOB communication is authentic: the adversary can only read and delay these OOB messages.

We emphasise that there are no true broadcast channels in our model. Although several wireless networks such as WLAN in ad hoc mode offer physical broadcast channels, there are no guarantees that the signal actually reaches all nodes. If we can guarantee this by physical means, then the authentication task becomes almost trivial. As different recipients can receive different broadcast messages, there is no difference between broadcasting and standard messaging except for efficiency. Similarly, broadcasting authenticated messages does not change the security analysis, although in practice, broadcasting can significantly reduce the necessary human interaction and make the protocol more user-friendly. For instance, a user entering the same PIN on each mobile device in a Bluetooth piconet is certainly a less demanding than using different PIN values. The same is true if we consider securing of VoIP-based conference calls: a participant giving the same value to all others has much less work than a participant giving a different value to each group member.

3.2 Idealised implementation

Many message authentication protocols are designed to meet complex security objectives. Therefore, it is often advantageous to fix the desired idealised implementation $\pi_\circ$ first and then define security of a protocol π through a specific game that directly quantifies how much the real execution diverges from the idealised implementation $\pi_\circ$.

As there are many possible ideal implementations, various specifications and security requirements for message authentication protocols form a complex taxonomy. For example, there are two-party and group authentication protocols. Also, a message authentication protocol can assure integrity of a single message or protect a combined input that is assembled by many participants.

In the simplest case, the set of participants is known ahead and the protocol must assure only the authenticity of transferred messages. The latter is true for many practical applications such as securing conference calls over VoIP, forming Bluetooth piconets and including devices to other wireless networks. Alternatively, the group can be formed dynamically based on the participation such as configuration of complex sensor networks. Then the protocol must additionally assure that honest participants also agree on the group description, i.e., received identities coincide.

All these security goals can be formalised by choosing an appropriate ideal world model, where a trusted third party $\mathcal{T}$ does all computations. More formally, assume that the trusted third party $\mathcal{T}$ can securely exchange messages with parties $\mathcal{P}_1, \ldots, \mathcal{P}_N$ and the ideal world adversary $\mathcal{A}^\circ$. Additionally, assume that a node label $\mathsf{id} \in \{1, \ldots, N\}$ uniquely determines the corresponding node $\mathcal{P}_{\mathsf{id}}$, i.e., a node label id can be treated as an network address. Then the ideal implementation $\pi_\circ$ can be formalised by specifying the behaviour of $\mathcal{P}_1, \ldots, \mathcal{P}_N$ and $\mathcal{T}$.

For example, consider an ordinary message authentication protocol, where a party $\mathcal{P}_{\mathsf{id}_1}$ wants to send a message m to a party $\mathcal{P}_{\mathsf{id}_2}$ in an authenticated way. The corresponding idealised implementation $\pi_\circ$ (depicted in Figure 2) models all attacks that cannot be avoided. Obviously, we cannot guarantee that the message m reaches the destination in the real world, since the

adversary can always drop all in-band messages. Secondly, the adversary can decide whether to corrupt the sender depending on the first in-band message which often reveals the input m. In other words, the adversarial corruption pattern can always depend on the message m. The ideal implementation of a two-party cross-authentication protocol, where parties want to exchange their inputs m_1 and m_2, is defined analogously.

Figure 2 The ideal unilateral authentication protocol

1. The party $\mathcal{P}_{\mathsf{id}_1}$ sends the message m first to $\mathcal{A}^\circ$ and then to $\mathcal{T}$.
2. The adversary $\mathcal{A}^\circ$ can then either halt the execution or not:
   - If $\mathcal{A}^\circ$ sends 0 to $\mathcal{T}$ then $\mathcal{P}_{\mathsf{id}_2}$ receives $\perp$ from $\mathcal{T}$.
   - If $\mathcal{A}^\circ$ sends 1 to $\mathcal{T}$ then $\mathcal{P}_{\mathsf{id}_2}$ receives m from $\mathcal{T}$.

As a second example, we present an idealised implementation for a group authentication protocol, where the set of participants is determined dynamically and the final outcome is combined from all inputs. More precisely, a Group Message Authentication (GMA) protocol for a group[1] $\mathcal{G} = \{\mathsf{id}_1, \ldots, \mathsf{id}_n\}$ works as follows. Each participant $\mathcal{P}_{\mathsf{id}}$, $\mathsf{id} \in \mathcal{G}$ starts with inputs m_{id} and ends with outputs $\mathcal{G}$ and $\boldsymbol{m}$, where $\boldsymbol{m} = (m_{\mathsf{id}_1}, \ldots, m_{\mathsf{id}_n})$. As a result, given $\mathcal{G}$ and $\boldsymbol{m}$ it is trivial to restore who participated in the protocol and what was the corresponding input.

Again, the corresponding idealised implementation (depicted in Figure 3) models all unavoidable attacks. In particular, note that a real world adversary can always control who joins the group by selectively blocking in-band messages. Also, note that we can obtain descriptions of different GMA protocols by modifying the ideal implementation. For example, if we drop the first step then we obtain a group authentication protocol without an initiator.

3.3 Stand-alone security

For clarity, we first consider security of message authentication protocols in the stand-alone model, where the adversary can attack only a single protocol instance. More precisely, we consider security against chosen inputs attacks in the common reference string model.

In the corresponding security game, the challenger first generates system wide public parameters $\mathsf{pk} \leftarrow \mathsf{setup}$ and sends them to the adversary $\mathcal{A}$. Then the adversary $\mathcal{A}$ can adaptively specify inputs for all parties $\mathcal{P}_1, \ldots, \mathcal{P}_N$ who can join the authentication protocol π. More precisely, a party $\mathcal{P}_i$ remains inactive until the adversary $\mathcal{A}$ specifies its input m_i. If $m_i = \perp$ then the party $\mathcal{P}_i$ refuses to join the protocol π. Otherwise, the party $\mathcal{P}_i$ joins the protocol π with the input m_i. The adversary $\mathcal{A}$ can also adaptively corrupt protocol participants. At the end of the execution, the challenger collects all outputs $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_N, \psi_{\mathrm{a}})$ and determines whether $\mathcal{A}$ succeeded in deception or not.

For simple protocols, it is easy to define deception by listing all invalid end configurations. However, such an approach quickly becomes tedious and technical for complex protocols. Hence, we use a generic approach and state that the adversary $\mathcal{A}$ *succeeds in deception* if one cannot achieve the same end configuration for honest participants in the ideal world. Formally, let $\mathcal{H} \subseteq \{1, \ldots, N\}$ be the set of non-corrupted participants and let $(m_i)_{i \in \mathcal{H}}$ be the corresponding inputs. Then the adversary $\mathcal{A}$ *fails in deception* if one can choose inputs $(\hat{m}_i)_{i \notin \mathcal{H}}$ for the corrupted participants such that the ideal world adversary $\mathcal{A}^\circ$ can achieve the same end configuration $(\psi_i)_{i \in \mathcal{H}}$ for the honest parties.

Definition 1: A message authentication protocol π is (t, ε)-*secure in the stand-alone model* if for any t-time real world adversary $\mathcal{A}$ the deception probability

$$\mathrm{Adv}_{\pi}^{\mathsf{forge}}(\mathcal{A}) = \Pr\left[\mathsf{pk} \leftarrow \mathsf{setup} : \mathcal{A} \text{ succeeds in deception}\right]$$

is bounded by $\mathrm{Adv}_{\pi}^{\mathsf{forge}}(\mathcal{A}) \leq \varepsilon$.

Let us now consider compatible real and ideal world adversaries that both specify the same inputs m_i and corrupt the same set of participants in the same order. To be punctual, we assume that the setup procedure is executed also in the ideal world[2] and the corresponding distributions of inputs and corrupted parties coincide for any value of public parameters pk. Let $\boldsymbol{\psi} = (\psi_i, \ldots, \psi_n, \psi_{\mathrm{a}})$ and $\boldsymbol{\psi}^\circ = (\psi_i^\circ, \ldots, \psi_n^\circ, \psi_{\mathrm{a}}^\circ)$ denote the corresponding output distributions. Then for any pair of compatible adversaries $(\mathcal{A}, \mathcal{A}^\circ)$ the statistical difference

Figure 3 Idealised implementation of a dynamic GMA protocol

1. An initiator node $\mathcal{P}_{\mathsf{id}_*}$ sends $(\mathsf{id}_*, m_{\mathsf{id}_*})$ first to $\mathcal{A}^\circ$ and then to $\mathcal{T}$.
2. The adversary adaptively determines the set of participants $\mathcal{G}$:
   (a) $\mathcal{A}^\circ$ sends an identity id to $\mathcal{T}$ who sends an invitation message to $\mathcal{P}_{\mathsf{id}}$.
   (b) $\mathcal{P}_{\mathsf{id}}$ can join the protocol by sending $(\mathsf{id}, m_{\mathsf{id}})$ first to $\mathcal{A}^\circ$ and then to $\mathcal{T}$.
   (c) Alternatively $\mathcal{P}_{\mathsf{id}}$ can send $\perp$. Then $\mathcal{P}_{\mathsf{id}}$ is not included to the group.
   (d) Steps (a)–(c) are repeated until $\mathcal{A}^\circ$ stops the group formation phase.
3. The adversary $\mathcal{A}^\circ$ can now either halt the execution or not:
   - If $\mathcal{A}^\circ$ sends 0 to $\mathcal{T}$ then all group members receive $\perp$ from $\mathcal{T}$.
   - If $\mathcal{A}^\circ$ sends 1 to $\mathcal{T}$ then all group members receive $(\mathcal{G}, \boldsymbol{m})$ from $\mathcal{T}$.

between the distributions ψ and ψ° is at least $\mathrm{Adv}_\pi^{\mathsf{forge}}(\mathcal{A})$. However, it is also straightforward to prove that this bound is optimal when all inputs are extractable from the outputs of honest participants. Namely, there exists a canonical ideal world adversary $\mathcal{A}^\circ$ with comparable running time such that the corresponding output distributions are $\mathrm{Adv}_\pi^{\mathsf{forge}}(\mathcal{A})$-close. Although we state the corresponding theorem only for group authentication protocols, it holds for all message authentication protocols considered in this paper.

Theorem 2: *If a group message authentication protocol π is (t,ε)-secure in the stand-alone model, then for any t-time real world adversary $\mathcal{A}$ there exist a compatible $t+\mathcal{O}(1)$-time ideal world adversary $\mathcal{A}^\circ$ such that the corresponding output distributions ψ and ψ° are ε-close.*

Proof: For the proof, we construct an universal interface $\mathcal{I}$ between the ideal world and the real world adversary $\mathcal{A}$, depicted in Figure 4. The interface $\mathcal{I}$ acts as a translation unit. It simulates real world execution to the adversary $\mathcal{A}$ and carries out the corresponding ideal world attack.

Note that the simulation of honest participants is straightforward, since honest participants forward their inputs m_i to the ideal world adversary $\mathcal{A}^\circ$ and public parameters pk are fixed by the challenger. Hence, the interface $\mathcal{I}$ can do all missing computations in behalf of honest parties. In particular, we can simulate the group formation, as the actions of the real world adversary uniquely determine when and which parties are included into the group. To service a corruption call, the interface just forwards the call to the ideal world and then adds all variables that are used in simulation to the released state.

At the end of simulation, the interface $\mathcal{I}$ internally obtains all outputs of honest parties and the adversary $\mathcal{A}$ submits also the remaining outputs. As the simulation is perfect, the corresponding output vector ψ coincides with the outputs obtained in the real execution. Now the interface can extract missing inputs $(\hat{m}_i)_{i\notin\mathcal{H}}$ from the outputs of honest parties and submit them to $\mathcal{T}$ as the inputs of corrupted participants. To be precise there are three possibilities. First, all honest parties can halt with $\perp$, then the interface $\mathcal{I}$ must send 0 to $\mathcal{T}$. Secondly, all outputs of honest participants provide the same missing inputs $(\hat{m}_i)_{i\notin\mathcal{H}}$. Then the interface $\mathcal{I}$ must send 1 to $\mathcal{T}$. Thirdly, the outputs of honest participants lead to different inputs $(\hat{m}_i)_{i\notin\mathcal{H}}$. Then the interface has failed. To conclude the ideal world attack, the interface forwards the outputs $(\psi_i)_{i\notin\mathcal{H}}$ and ψ_a to the challenger.

Evidently, the failure probability must be less than ε or otherwise the protocol cannot be (t,ε)-secure. Therefore, the compound adversary $\mathcal{I}\langle\mathcal{A}\rangle$ consisting of $\mathcal{I}$ and $\mathcal{A}$ has indeed the desired properties. □

Note that stand-alone security model covers only the case where no other protocols are executed together with the protocol π. In particular, it is not clear whether a concurrent execution of several different protocol instances remains secure. We will return to this issue in Section 6 and show that such concurrent compositions remain secure if some natural assumptions are satisfied.

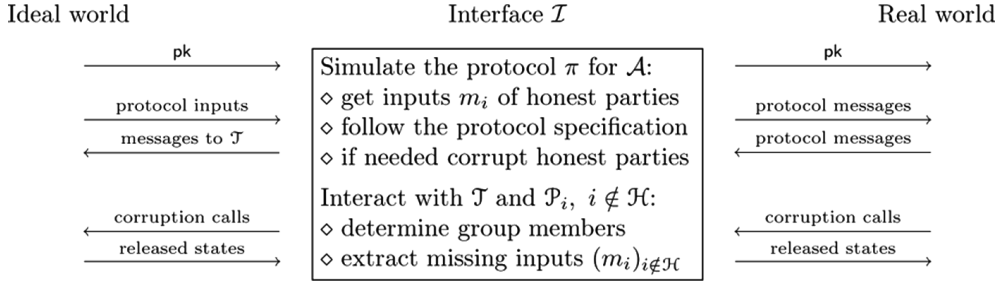
4 Two-party authentication protocols

Message authentication protocols either use pre-shared information such as long-term secret keys or alternatively rely on limited authentic OOB communication. In the following, we give a systematised treatment of common user-aided data authentication protocols, where a user is allowed to transfer ℓ-bits of information over authentic OOB channel. Such protocols are normally used to bootstrap other more complex authentication methods.

For instance, many practical communication protocols like SSH, PGP and GPG send public key over the insecure channel and then validate their integrity with OOB communication. In the most naive setting, the fingerprint of a transferred message is just its hash value. The corresponding protocol was first formalised by Balfanz et al. (2002). The protocol has two main drawbacks. First, an adversary can always conduct offline attacks by seeking messages with coinciding fingerprints. Secondly, due to the birthday paradox one can find collisions in time $\Theta(2^{\ell/2})$ and thus the fingerprint must be several hundred bits long. Hence, the solution is not user-friendly despite the trials to transform the tedious hexadecimal representation into a nice word-based representation.

The possibility of offline attacks can be defeated with message salting. As a result, fingerprints vary even for a fixed message and the corresponding generic collision attacks are guaranteed to run in time $\Theta(2^\ell)$. Pasini and Vaudenay were the first to design such a protocol (Pasini and Vaudenay, 2006a). They proposed to send commitment decommitment pair $(c,d) \leftarrow \mathsf{com}_{\mathsf{pk}}(m)$

Figure 4 The canonical interface for the real world adversary

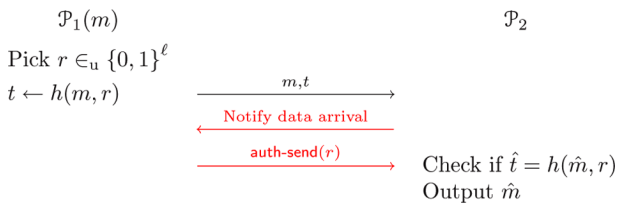


instead of the message m and use the hash value $h(c)$ as a fingerprint. Of course, using a commitment scheme as a mechanism for message salting is not the most optimal design choice. Therefore, several authors have proposed alternative methods that rely only on specific properties of cryptographic hash functions, see Mashatan and Stinson (2006) and Reyhanitabar et al. (2007). Note that a commitment scheme can be built just using a hash function and so the protocol from Pasini and Vaudenay becomes also simple.

However, such non-interactive authentication protocols are still vulnerable against online computation attacks when the fingerprint is short enough. In many practical settings, users are not willing to transfer OOB messages that are more than 4–6 decimal digits long. Consequently, generic collision finding strategies with complexity $\Theta(2^{\ell})$ become feasible. To overcome this shortcoming, Gehrmann, Mitchell, and Nyberg proposed three protocols known as the MANA family Gehrmann et al. (2004). Although the MANA I and MANA II protocols are known to be secure against unbounded adversaries, the corresponding deception probability is sub-optimal. Namely, if a user is willing to transfer ℓ-bit OOB messages, then there are still attacks with success probability at least $2^{-\ell/2}$.

Vaudenay noticed this drawback and proposed a protocol (Vaudenay, 2005) that achieves the optimal deception bound $2^{-\ell}$. This paper was soon followed by slew of works (Pasini and Vaudenay, 2006b; Laur and Nyberg, 2006; Laur and Pasini, 2008) that described asymptotically optimal protocols for many other settings. In a certain sense, all these solutions are just enhancements of the MANA I and the MANA II protocols. The corresponding generic technique that substitutes a private pre-shared secret with a public OOB message is depicted in Figure 5.

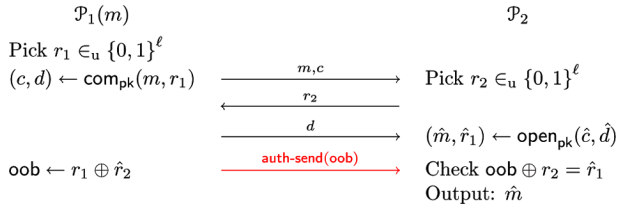
Figure 5 The simplified MANA II protocol (see online version for colours)



The protocol uses a (universal) hash function to protect the integrity of the message m. Due to the protocol structure, the adversary must deliver a possibly altered message $\hat{m}$ and a digest $\hat{t}$ before the key r is released. Hence, the setting is equivalent to the classical attack scenario, where parties $\mathcal{P}_1$ and $\mathcal{P}_2$ pre-share the secret key r. Consequently, standard lower bounds on deception probabilities derived by Simmons are still adequate, see Simmons (1984) and Maurer (2000). Namely, the hash function must satisfy contradictory security requirements in order to preserve security against substitution or impersonation attacks. As a result, the deception probability is guaranteed to be at least $2^{-\ell/2}$. Secondly, the protocol contains an extra OOB message that prevents too early key release.

The SAS protocol proposed by Vaudenay is free of these limitations although its structure is very similar to the simplified MANA II protocol, see Figure 6. Indeed, note that the committed pair (m, r_1) corresponds to the simplest message authentication code that achieves optimal security against impersonation attacks but is completely insecure against substitution attacks. However, the commitment creates an extra bond between m and r_1 that makes simple substitution attacks infeasible. The second in-band message has the same function as the notification message in the simplified MANA II protocol.

Figure 6 The Vaudenay SAS protocol (see online version for colours)



Indeed, note that the corresponding key r_1 is transferred over the OOB channel in an encrypted form $r_1 \oplus \hat{r}_2$. Hence, the adversary cannot predict the decrypted value oob $\oplus\, r_2$ before getting the secret key r_2. Consequently, the adversary can either learn (m, r_1) and carry out an impersonation attack or alternatively try to alter the commitment. The latter is destined to fail when the commitment is assumed to be non-malleable. For the completeness, we give also the formal proof, since the original paper (Vaudenay, 2005) required more exotic properties from the commitment scheme, see Appendix A.

Theorem 3: *For any t there exists $\tau = t + \mathcal{O}(1)$ such that if the commitment scheme is $(2\tau, \varepsilon_{\mathrm{b}})$-binding and $(\tau, \varepsilon_{\mathrm{nm}})$-non-malleable, then the SAS protocol with ℓ-bit* oob *is $(t, 2^{-\ell} + \varepsilon_{\mathrm{b}} + \sqrt{\varepsilon_{\mathrm{b}}} + \varepsilon_{\mathrm{nm}})$-secure in the stand-alone model.*

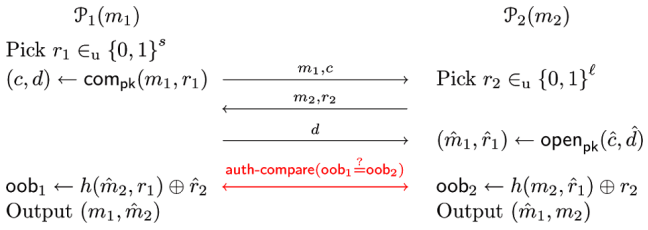
As a final detail, note that the commitment scheme is only used to bind the message together with the digest. Therefore, one can use more efficient tag-based commitments that do not hide the value of m but still bind the pair m and r_1 together. In particular, signature schemes can be used as tag-based commitments (Vaudenay, 2005).

The SAS protocol provides only unilateral message authentication and thus can be used to secure the transfer of public keys. However, common key agreement protocols such as the Diffie-Hellman protocol consist of many moves. Now if one uses the SAS protocol for each message, the cumulative security drop is significant as the deception probability is non-negligible. Also, the amount of user interaction increases significantly. Alternatively, we could use the SAS protocol to authenticate the entire transcript of the key agreement protocol. The latter leads to the generic cross-authentication technique described below.

Any unilateral message authentication protocol can be turned into a cross-authentication protocol with a cost of an extra move. Namely, the recipient $\mathcal{P}_2$ must send its input m_2 to $\mathcal{P}_1$. Now if $\mathcal{P}_1$ authentically transfers $(m_1, \hat{m}_2)$ to $\mathcal{P}_2$, then $\mathcal{P}_2$ can additionally verify that m_2 has been correctly transferred, i.e., $\hat{m}_2 = m_2$. Thus, a successful completion of the authentication protocol assures that both parties have coinciding outputs (m_1, m_2).

Hence, it makes sense to design cross-authentication protocols that make at most three moves over the insecure channel. Two such protocols were almost simultaneously proposed by Pasini and Vaudenay (2006b) and by Laur and Nyberg (2006). These protocols send a message digest $h(m, r)$ over the OOB channel instead of the hash key r. Since the hash key can be arbitrarily long, one can bypass Simmons's bounds and achieve the optimal deception probability. Nevertheless, we still have to guarantee that the adversary has no access to the key r before both parties have acquired the common output. Such structural restrictions are enforced by clever use of commitments. The corresponding methodology is particularly apparent in the optimised SAS protocol proposed by Pasini and Vaudenay (2006b) as depicted in Figure 7.

Figure 7 The optimised SAS-MCA protocol (see online version for colours)

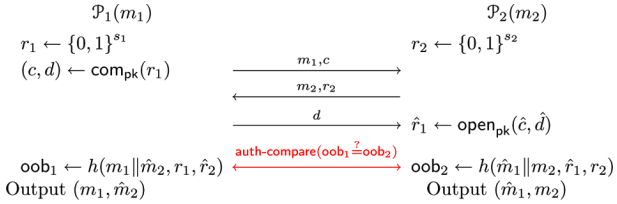


For a moment, assume that m_1 is fixed, i.e., we have an unilateral message authentication protocol for m_2. Then the usage of one-time pad $h(\hat{m}_2, r_1) \oplus \hat{r}_2$ assures that the adversary cannot succeed unless he transfers the commitment $\hat{c}$ before the decommitment d is released. But if the adversary just forwards c, then we arrive at the standard XOR-universality game, where the adversary must find $m_2 \neq \hat{m}_2$ such that $h(m_1, r_1) \oplus h(\hat{m}_2, r_1) = r_2 \oplus \hat{r}_2$ for an unknown key $r_1 \in_{\mathrm{u}} \{0,1\}^s$. Alternatively, the adversary can try to alter the non-malleable commitment but this is guaranteed to fail. For the same reason, the message m_1 is also guaranteed to reach $\mathcal{P}_2$ without modifications. As a result, we can easily establish the following security guarantee, see Appendix B for the proof.

Theorem 4: *Assume that the hash function h is ε_{r}-almost regular and ε_{u}-almost XOR-universal. Then for any t there exists $\tau = t + \mathcal{O}(1)$ such that if the commitment scheme is $(\tau, \varepsilon_{\mathrm{h}})$-hiding, $(2\tau, \varepsilon_{\mathrm{b}})$-binding and $(\tau, \varepsilon_{\mathrm{nm}})$-non-malleable, then the optimised SAS-MCA protocol with ℓ-bit* oob *is $(t, \max\{\varepsilon_{\mathrm{r}}, \varepsilon_{\mathrm{u}}\} + \varepsilon_{\mathrm{h}} + \varepsilon_{\mathrm{b}} + \sqrt{\varepsilon_{\mathrm{b}}} + \varepsilon_{\mathrm{nm}})$-secure in the stand-alone model.*

The MANA IV protocol (see Figure 8) proposed by Laur and Nyberg (2006) also uses commitments to temporarily hide hash keys. But differently from the SAS protocol family, the message pair (m_1, m_2) is directly authenticated with the hash function. Again, the protocol structure guarantees that the adversary cannot succeed if messages are transferred abnormally, i.e., $\hat{m}_2, \hat{r}_2$ arrive before $\hat{m}_1, \hat{c}$ or $\hat{d}$ is received before $\hat{m}_2, \hat{r}_2$. Now for the normal runs, the adversary has to fix messages $\hat{m}_1, \hat{m}_2$ before both sub-keys r_1 and r_2 become public. As a result, information theoretical properties of the hash function are sufficient to guarantee authenticity. See the paper Laur and Nyberg (2006) for the formal proof.

Figure 8 The MANA IV protocol (see online version for colours)



Theorem 5: *Assume that the hash function h is ε_{r}-almost regular w.r.t. sub-keys and ε_{u}-almost universal w.r.t. the sub-key r_1. Then for any t there exists $\tau = t + \mathcal{O}(1)$ such that if the commitment scheme is $(\tau, \varepsilon_{\mathrm{h}})$-hiding, $(2\tau, \varepsilon_{\mathrm{b}})$-binding and $(\tau, \varepsilon_{\mathrm{nm}})$-non-malleable, then the MANA IV protocol with ℓ-bit* oob *is $(t, \max\{\varepsilon_{\mathrm{r}}, \varepsilon_{\mathrm{u}}\} + 2\varepsilon_{\mathrm{h}} + \varepsilon_{\mathrm{b}} + \sqrt{\varepsilon_{\mathrm{b}}} + \varepsilon_{\mathrm{nm}})$-secure in the stand-alone model.*

Finally, observe that protocols with an optimal deception bound utilise similar techniques. First, all of them use one-time pad encryption to assure that the adversary preserves the temporal order between protocol messages. Secondly, the commitment scheme is used as an additional measure against substitution attacks. Thirdly, it seems that there are no other designs patterns that could overcome the shortcomings of the MANA II protocol.

5 Group authentication protocols

The methodology presented in the previous section can be naturally extended to group settings. However, there are some important differences. First, it is much more difficult to assure proper temporal order for send and receive events, since there are more events to be synchronised. Secondly, the set of participants might be determined *dynamically* during the protocol execution based on participation. Hence, we must also authenticate the group description.

Note that an authenticated broadcast primitive is sufficient for group message authentication. Namely, participants $\mathcal{P}_i$ can first send all messages m_i to a leader $\mathcal{P}_{\mathsf{id}_*}$ who then uses the authentic broadcast primitive to transfer the gathered input $\boldsymbol{m}_* = (m_i)_{i \in \mathcal{G}}$ and the group description $\mathcal{G}$ to all participants. Next, all

participants $\mathcal{P}_i$, $i \in \mathcal{G}$ verify that received message $\hat{\boldsymbol{m}}_*$ is consistent with their input m_i. The protocol is halted if a complaint is raised over the OOB channels. The authenticated broadcast primitive itself can be achieved running several unilateral authentication protocols in parallel. However, the latter automatically increases the number of different OOB messages. Alternatively, we can design specific protocols, where the same OOB message is transferred to all group members. Although this does not formally decrease the amount of OOB communication, it makes the protocol more user friendly. The first such broadcast protocol was sketched in Valkonen et al. (2006). The corresponding Group-MANA protocol is a simple extension of the MANA IV protocol. However, we do not discuss it here for two reasons. First, the SAS-GMA protocol (depicted in Figure 9) is more round efficient. Second, the corresponding security proof is rather technical and gives no additional insight.

The SAS-GMA protocol was designed by Laur and Pasini to achieve group message authentication directly, see Laur and Pasini (2008). As the SAS-GMA protocol is symmetric, Figure 9 only specifies the behaviour of a single party $\mathcal{P}_i$ who wants to participate in the protocol. Here $\hat{\mathcal{G}}_i$ denotes the group of participants who joined $\mathcal{P}_i$ during the first round before the timeout. Of course, if the group $\hat{\mathcal{G}}_i$ is known beforehand then $\mathcal{P}_i$ can wait until all other group members have sent their first messages. For clarity, variables $\hat{m}_{ji}, \hat{c}_{ji}, \hat{d}_{ji}$ denote the values from $\mathcal{P}_j$ that are received by $\mathcal{P}_i$. The output vector $\hat{\boldsymbol{m}}_i = (\hat{m}_{ji})$ and the sub-key vector $\hat{\boldsymbol{r}}_i = (\hat{r}_{ji})$ are ordered w.r.t. sender identities. To be exact, $\hat{m}_{ii} = m_i$, $\hat{r}_{ii} = r_i$ and j ranges over $\hat{\mathcal{G}}_i$. Also note that (i, r_i) and $(\hat{\mathcal{G}}_i, \hat{\boldsymbol{m}}_i)$ are shorthands for binary strings that uniquely encode the corresponding elements. Finally, we assume that a participant $\mathcal{P}_i$ halts if there is any hint of an attack:

- some group member halts;
- there are duplicates $(j, \hat{m}_{ji}, \hat{c}_{ji}) \neq (j, \hat{m}'_{ji}, \hat{c}'_{ji})$;
- a sub-key is invalid: $(j, \star) \neq \mathsf{open}_{\mathsf{pk}}(\hat{c}_{ji}, \hat{d}_{ji})$;
- some OOB messages do not match: $\mathsf{oob}_i \neq \mathsf{oob}_j$.

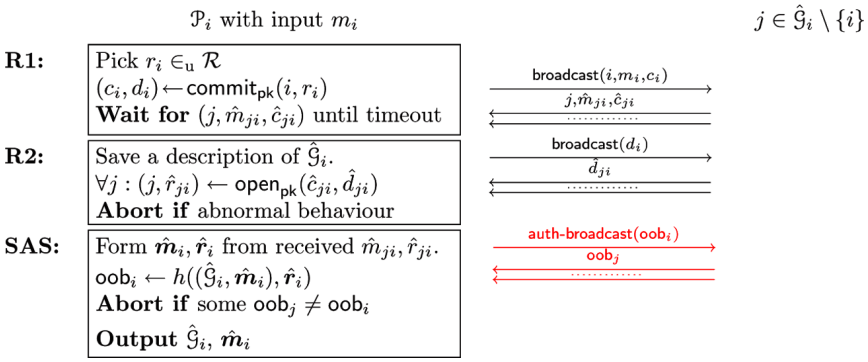
The SAS-GMA protocol does not directly force proper order of send and receive events, since the latter is extremely difficult to achieve in the group setting. Instead, the protocol uses a commitment scheme to temporarily hide many sub-keys r_i of the hash function. As a result, the adversary must deliver the data $\hat{\boldsymbol{m}}_i$ to the node $\mathcal{P}_i$ before the sub-key r_i is released. Now if the hash function is both almost universal and almost regular w.r.t. all sub-keys, then all impersonation and substitution attacks are guaranteed to fail unless the adversary alters commitments. On the other hand, non-malleability of a commitment scheme together with almost regularity defeats all attacks that use altered commitments. The corresponding formal proof is given in the paper (Laur and Pasini, 2008).

Theorem 6: *Let n be the maximal size of the group $\mathcal{G}$. Assume that the hash function h is ε_{u}-almost universal w.r.t. all sub-key pairs and ε_{r}-almost regular w.r.t. all sub-keys. Then for any t there exists $\tau = t + \mathcal{O}(1)$ such that if the commitment scheme is $(\tau, \varepsilon_{\mathrm{b}})$-binding and $(\tau, \varepsilon_{\mathrm{nm}})$-non-malleable, then the SAS-GMA protocol with ℓ-bit* oob *is $(t, \max\{\varepsilon_{\mathrm{u}}, \varepsilon_{\mathrm{r}}\} + \varepsilon_{\mathrm{b}} + n \cdot \varepsilon_{\mathrm{nm}})$-secure in the stand-alone model.*

6 Security in complex settings

The stand-alone security model is adequate only if a protocol is executed in isolation. This assumption is rarely fulfilled in practice, as protocols are often executed concurrently to complete more elaborate tasks. In such settings, stand-alone security guarantees are commonly insufficient, as the adversary can utilise external information that leaks from the other protocols. For example, the adversary may repeat or swap messages when several instances of the same protocol are executed at the same time. Bellare and Rogaway were the first to define a formal attack model (Bellare and Rogaway, 1993, 1995) that considers attacks against several instances of the same authentication protocol. However, the Bellare-Rogaway model does not cover the cases when the authentication protocol is executed together with other kind of protocols. In the following, we prove that all stand-alone secure authentication protocols are universally composable. In other words, an authentication protocol π always preserves security in a *computational context* $\varrho\langle\cdot\rangle$ that uses the protocol π in a black-box way, i.e., the context $\varrho\langle\cdot\rangle$

Figure 9 The SAS-GMA protocol (see online version for colours)



provides only the inputs and uses only the outputs of the authentication protocol π.

We emphasise that universal composability does not automatically guarantee security in the Bellare-Rogaway model. The main cause follows from the fact that all classical authentication protocols rely on a trusted setup procedure π_{st} that generates long-term secrets. As a result, the universal composability guarantees security only if all protocol instances use independently generated long-term secrets. The Bellare-Rogaway model considers a setting where all protocol instances share the same long-term secrets and thus universal composability might be insufficient. Obviously, these security notions coincide if we can guarantee security in the stand-alone model a without trusted setup. The latter makes user-aided message authentication protocols special.

6.1 Universal composability

As emphasised above, protocols are seldomly executed in isolation. Indeed, a protocol π is often only a small part of the entire computational procedure also known as computational context. Now if a context $\varrho\langle\cdot\rangle$ has only black-box access to the protocol, we can freely use different protocols as long as they implement the same functionality. In particular, we can compare the behaviour of the real and ideal implementations π and $\pi_\circ$. To be precise, we must compare the corresponding compound protocols $\varrho\langle\pi\rangle$ and $\varrho\langle\pi_\circ\rangle$.

Definition 7: Let $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_m, \phi_{\mathrm{a}})$ denote the inputs of the participants $\mathcal{P}_1, \ldots, \mathcal{P}_m$ and the adversary $\mathcal{A}$ at the beginning of the context $\varrho\langle\cdot\rangle$. Similarly, let the vectors $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_m, \psi_{\mathrm{a}})$ and $\boldsymbol{\psi}^\circ = (\psi_1^\circ, \ldots, \psi_m^\circ, \psi_{\mathrm{a}}^\circ)$ denote the outputs of the compound protocols $\varrho\langle\pi\rangle$ and $\varrho\langle\pi_\circ\rangle$. Then a protocol π is $(t_{\mathrm{re}}, t_{\mathrm{id}}, t_\varrho, \varepsilon)$-*universally composable* if for any input distribution $\boldsymbol{\phi} \leftarrow \mathfrak{D}$, for any t_ϱ-time computational context $\varrho\langle\cdot\rangle$ and for any t_{re}-time adversary $\mathcal{A}$ against the protocol $\varrho\langle\pi\rangle$, there exists a t_{id}-time adversary $\mathcal{A}^\circ$ against $\varrho\langle\pi_\circ\rangle$ such that the statistical difference between the output distributions of $\boldsymbol{\psi}$ and $\boldsymbol{\psi}^\circ$ is at most ε.

Definition 7 remains ambiguous unless we completely specify the execution and communication model. In the following, we consider the classical setting, where the adversary has full control over the protocols scheduling and message delivery. Namely, the execution of a protocol is divided into fine-grained micro-rounds. All parties are initially inactive except the adversary. The adversary can activate other participants so that only one of them is active in each micro-round. During a micro-round, the active participant can either read one incoming message or compose a single outgoing message. After that the party is suspended and the control goes back to the adversary who can choose next party for activation. All in-band messages are routed through the adversary who can delay, read, delete and insert messages. The OOB communication is authentic but the adversary can still read, delay and reorder messages. The execution ends when all participants have halted. See the manuscript (Canetti, 2000) for detailed discussion and for further references.

Finally, we remark that the approach outlined above corresponds to the most intuitive formalisation (Lindell, 2003) of universal composability but there are several more popular alternatives such as the treatments (Canetti, 2001; Pfitzmann and Waidner, 2001).

6.2 Protocols with shared setup

Many protocols rely on pre-shared information like long-term secret keys or certificate chains. Such protocols can be divided into two phases. In the first phase, a trusted dealer creates and securely distributes the necessary pre-shared data. The second phase corresponds to the actual execution of the protocol. Hence, the protocol π itself is a pair of sub-protocols $(\pi_{\mathrm{st}}, \pi_{\mathrm{ne}})$ where π_{st} corresponds to the *trusted setup* and π_{ne} corresponds to the actual execution. Normally, we want to reuse pre-shared data and thus different protocols must share the same setup phase π_{st}. As a result, messages from different protocols become correlated and this creates new attack opportunities. The security model proposed by Bellare and Rogaway formalises the corresponding threats for authentication protocols, see the papers (Bellare and Rogaway, 1993, 1995).

Differently from the stand-alone model, the adversary $\mathcal{A}$ can simultaneously attack many protocol instances $\pi_{\mathrm{ne}}^{(1)}, \ldots, \pi_{\mathrm{ne}}^{(q)}$ that share the same setup phase π_{st}. More formally, the adversary $\mathcal{A}$ can adaptively launch new protocol instances $\pi_{\mathrm{ne}}^{(i)}$ by specifying the set of participants $\mathcal{G}^{(i)}$ and the corresponding inputs $\boldsymbol{m}^{(i)}$. The adversary $\mathcal{A}$ succeeds in deception if at least one protocol instance ends with successful deception.

Definition 8: A message authentication protocol π is (t, q, ε)-*strongly self-composable* if any t-time adversary $\mathcal{A}$ that can launch up to q protocol instances $\pi^{(i)}$ with $i \in \{1, \ldots, q\}$ succeeds in deception with probability less than ε.

Note that a shared setup phase may weaken protocol instances even if we reuse only public parameters. Hence, we must prove that security in the common reference string model guarantees security in the Bellare-Rogaway model.

6.3 Composability guarantees

Regardless of the desired idealised implementation it is straightforward to prove that all message authentication protocols are universally composable if they are secure in the stand-alone model. For brevity, we prove the corresponding result only for group authentication protocols and discuss the limitations of this proof technique below.

Theorem 9: *Let π be a (t, ε)-secure group message authentication protocol. Then there are constants*

c_1, c_2 such that the protocol is $(t_{\mathrm{re}}, t_{\mathrm{id}}, t_\varrho, \varepsilon)$-universally composable whenever $t_{\mathrm{id}} \geq c_1 \cdot t_{\mathrm{re}}$ and $t_{\mathrm{re}} + t_\varrho \leq t - c_2$.

Proof: Let $\varrho\langle\cdot\rangle$ be a t_ϱ-time computational context and let $\mathcal{A}$ be a t_{re}-time adversary against the compound protocol $\varrho\langle\pi\rangle$. Then for the proof we construct an efficient interface $\mathcal{I}_*$ between the real world adversary $\mathcal{A}$ and the ideal world protocol $\varrho\langle\pi_\circ\rangle$. Now note that the interface $\mathcal{I}$ depicted in Figure 4 and described in the proof of Theorem 2 is sufficient for this purpose if we can separate protocol and non-protocol messages. The corresponding construction is depicted in Figure 10. That is, we direct non-protocol messages past the interface $\mathcal{I}$. To be precise, we must guarantee that the simulation is perfect, i.e., the adversary sees the messages in the same order as in the real execution. Hence, we must additionally assume that there is a general (possibly dynamic) scheduling policy that uniquely determines in which order an honest participant $\mathcal{P}_i$ outputs protocol and non-protocol messages. Secondly, the corruption calls are handled by the interface $\mathcal{I}$ that corrupts the honest participant and adds the variables used in simulation to the state of released participant.

Now it is straightforward to verify that the simulation of the protocol is perfect and that there can be a discrepancy between the real and ideal world outputs ψ and ψ° only if the adversary $\mathcal{A}$ succeeds in deception. The corresponding deception probability must be less than ε or otherwise the real world adversary $\mathcal{A}$ together with the context $\varrho\langle\cdot\rangle$ forms a new stand-alone adversary $\mathcal{A}^*$ that achieves $\mathrm{Adv}_\pi^{\mathsf{forge}}(\mathcal{A}^*) > \varepsilon$. The latter leads to a contradiction, since the running time of $\mathcal{A}^*$ is $t_{\mathrm{re}} + t_\varrho + \mathcal{O}(1)$. Now the claim follows, as the overhead in the simulation is constant. □

As a first limitation, note that the interface $\mathcal{I}$ may completely fail if protocols share the same trusted setup π_{st}. For obvious reasons, such failures are caused by protocols $\pi_{\mathrm{ne}}^{(1)}, \ldots, \pi_{\mathrm{ne}}^{(q)}$ that share the long-term secrets. If the trusted setup π_{st} is run independently from the interface $\mathcal{I}$, then it does not know the corresponding long-term secrets and cannot simulate the execution of honest parties. Alternatively, if the setup π_{st} is a part of the interface $\mathcal{I}$, then we can replace only a single protocol instance $\pi_{\mathrm{ne}}^{(i)}$ with the ideal implementation. After that the corresponding adversary $\mathcal{I}\langle\mathcal{A}\rangle$ knows all long-term secrets and all other protocol instances $\pi_{\mathrm{ne}}^{(1)}, \ldots, \pi_{\mathrm{ne}}^{(q)}$ become insecure. Hence, one needs more elaborate security proofs for all authentication protocols that are based on long-term secrets. The latter is expected result, as some of these protocols are known to be secure in the stand-alone model but insecure in the Bellare-Rogaway model.

However, if the trusted setup phase generates only public values, then these problems disappear, as the knowledge of public parameters is sufficient to simulate the behaviour of honest parties. In particular, the adversary $\mathcal{A}^*$ is still a valid stand-alone standalone adversary and the proof of Theorem 9 still holds.

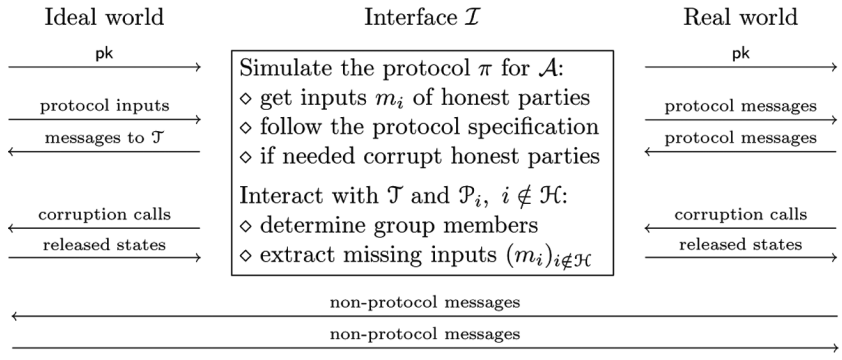
Corollary 10: *Let π be a group authentication protocol that is (t, ε)-secure in the common reference string model. Then there are constants c_1, c_2 such that the protocol is $(t_{\mathrm{re}}, t_{\mathrm{id}}, t_\varrho, \varepsilon)$-universally composable whenever $t_{\mathrm{id}} \geq c_1 \cdot t_{\mathrm{re}}$ and $t_{\mathrm{re}} + t_\varrho \leq t - c_2$ even if the protocol shares the setup phase with other protocols.*

This result represents the main technical difference between classical and user-aided data authentication. User-aided data authentication protocols are not based on long-term secrets and thus they remain secure in any computational context. Classical authentication protocols are also universally composable as long as secret keys are used only once. If we want to reuse secret keys, then we must prove security in the Bellare-Rogaway model. The corresponding proof implicitly shows that we can treat $\pi_{\mathrm{st}}, \pi_{\mathrm{ne}}^{(1)}, \ldots, \pi_{\mathrm{ne}}^{(q)}$ as single complex multi-round authentication protocol and replace it with the corresponding ideal implementation. The formal proof is analogous to the proof of Theorem 9 and thus we can use classical authentication protocols in any computational context as soon as they are secure in the Bellare-Rogaway model.

6.4 Message identification

As a second subtle detail, note that the proof outlined above is valid only if the interface $\mathcal{I}_*$ can correctly separate protocol messages from non-protocol messages. Otherwise, messages may become switched between different protocols and the corresponding *synchronisation errors* can cause arbitrary failures. To avoid such subtle

Figure 10 The canonical interface for complex settings



issues, theoretical treatments often assume that each message contains a specific tag that uniquely determines the corresponding protocol instance (Canetti, 2000). Consequently, we can always avoid synchronisation errors for in-band communication without excessive performance penalties. However, the latter is not true for OOB messages, since we do not want to increase the amount of authenticated communication. Of course, these tags can be dropped as long as they can be restored from the available information, i.e., the source and destination information uniquely determines the corresponding protocol instance. Thus, a user-aided data authentication protocol remains universally composable if the following restrictions hold:

$\mathcal{R}_1$: Randomness used in the protocol instance is freshly generated.

$\mathcal{R}_2$: The outputs are never used before all parties reach accepting state.

$\mathcal{R}_3$: All group members have different identities, i.e., $\mathcal{G}$ is indeed a set.

$\mathcal{R}_4$: The OOB messages determine a unique protocol instance.

Restrictions $\mathcal{R}_1 - \mathcal{R}_3$ are natural requirements and we can force the restriction $\mathcal{R}_4$ if we guarantee that no more than one protocol instance for the same group is run at the same time. The latter is a relatively mild limitation, since two or more parallel instances of an authentication protocol can be replaced with a single protocol instance. Hence, several papers (Laur and Nyberg, 2006; Laur and Pasini, 2008) have just postulated the usage restrictions $\mathcal{R}_1 - \mathcal{R}_4$ and have not studied the maximal damage caused by synchronisation errors. For the two party protocols, the corresponding extra advantage has been estimated in the papers (Vaudenay, 2005; Pasini and Vaudenay, 2006b).

Theorem 11: *Let π be a (t, ε)-secure cross authentication protocol between $\mathcal{P}_1$ and $\mathcal{P}_2$. If $\mathcal{P}_1$ launches up to q_1 and $\mathcal{P}_2$ up to q_2 concurrent instances of the protocol π, then the deception probability can increase up to $q_1 q_2 \cdot \varepsilon$.*

Proof sketch: Note that a successful deception pairs an instance launched by $\mathcal{P}_1$ and an instance launched by $\mathcal{P}_2$. Let ε_{ij} denote the probability that the first deception event happens for the ith instance launched by $\mathcal{P}_1$ and for the jth instance launched by $\mathcal{P}_2$. Then the overall deception probability $\mathrm{Adv}^{\mathsf{forge}}_{\pi,\ldots,\pi}(\mathcal{A})$ is just a sum of all ε_{ij}. Hence, one can create a stand-alone adversary with the success probability $\mathrm{Adv}^{\mathsf{forge}}_{\pi}(\mathcal{A}) \geq \frac{1}{q_1 q_2} \cdot \mathrm{Adv}^{\mathsf{forge}}_{\pi,\ldots,\pi}(\mathcal{A})$ by simulating all protocol instances except for a random instance pair that is substituted with the challenge instance. The claim follows. □

As a protocol with an optimal deception bound has uniformly distributed OOB message (Laur and Nyberg, 2006), the bound is also quite tight. A similar result holds also for the group setting. However, there the decrease in security level is much steeper, since the number of potential matches is significantly bigger. Therefore, it is much wiser to follow the restriction $\mathcal{R}_4$.

7 Key agreement protocols

The main application of user-aided message authentication protocols is to protect key agreement protocols against active attacks. In the following, we outline how to achieve this goal with minimal amount of user interaction.

Note that there is an inevitable trade-off between the security and usability. In many practical applications, users are willing to transfer only OOB messages that are up to six digits long. Hence, such authentication mechanisms can be bypassed with probability 2^{-20}. On the other hand, 2^{-20} is also the probability of not noticing an active attack. The latter is small enough to demotivate most attackers and thus the subjective security level can be much higher. For instance, if the probability of an active attack is below 10^{-6} then the achievable security level is 2^{-40}.

Of course, true cryptographic security can be achieved only with sufficiently long OOB messages. Hence, it is important to minimise the total amount of manually authenticated communication. In particular, it should be easy to exclude corrupted nodes from a group without transferring any additional OOB messages.

7.1 Formal definitions

A *group key agreement* protocol π between n participants $\mathcal{G} = \{\mathsf{id}_1, \ldots, \mathsf{id}_n\}$ starts with no input, is independent from the current state, and outputs $\mathcal{G}$ and a shared common secret key $\mathsf{sk} \in_{\mathrm{u}} \mathcal{K}$.

Definition 12: A group key agreement protocol π is (t, ε)-*immune against active attacks* if for any t-time adversary $\mathcal{A}$ that can choose a group $\mathcal{G} = \{\mathsf{id}_1, \ldots, \mathsf{id}_n\}$ then the probability that uncorrupted parties do not detect active attack is less than ε.

Obviously, any key agreement protocol that is (t, ε_1)-immune against active attacks and (t, ε_2)-secure against passive attacks is also $(t, \varepsilon_1 + \varepsilon_2)$-secure, as long as both definitions are given in the same attack model. In particular, we can construct also universally composable user-aided key agreement protocols as long as the underlying key agreement protocol is universally composable against passive attacks.[3] However, stand-alone security is sufficient for many practical settings, since the key agreement protocols are often executed in isolation to set up the communication network.

7.2 General construction

The most straightforward way to achieve (t, ε)-immunity against active attacks is to authenticate the entire protocol transcript. Namely, participants must first execute the group key agreement protocol and then

use the group message authentication protocol to verify that all transferred message were unaltered. A naive implementation, where protocols are executed sequentially, adds three extra rounds to the key agreement protocol. However, since message authentication protocols are universally composable, we can execute them in parallel and save two messages for two-party and one complete round for group protocols. It is also possible to fuse both protocols more tightly and thus obtain a more efficient protocols, see Laur and Nyberg (2006).

For two-party protocols, it is reasonable to combine the Diffie-Hellman key agreement protocol with one of the message cross-authentication protocols discussed in Section 4. The Burmester-Desmedt (BD) key agreement protocol (Desmedt and Burmester, 1994) is a suitable starting point for group settings, since it is provably secure against passive attacks (Burmester and Desmedt, 2005). Though the Burmester-Desmedt key agreement protocol is a generalisation of the Diffie-Hellman key agreement protocol, it can also be generalised for other two-party key agreement protocols, see the compiler of Just and Vaudenay (1996). For simplicity, consider a group of n participants $\mathcal{P}_0, \ldots, \mathcal{P}_{n-1}$ arranged in a ring. The Burmester-Desmedt protocol is depicted on Figure 11. It consists of two rounds over an authenticated channel, while most of the schemes requires $\mathcal{O}(n)$ rounds. Here, let g be a generator of a q-element secure Diffie-Hellman Decision Group. At the end of the protocol, each participant $\mathcal{P}_i$ obtains the same secret key $\mathsf{sk} = g^{k_1k_2+k_2k_3+\cdots+k_nk_1}$.

In many practical settings, we must be able to expel group members that behave maliciously. Ideally, this operation should not use additional OOB messages. Consequently, a simple key agreement protocol is not suitable for our needs, since a shared key gets compromised as soon as a group member gets corrupted. To avoid this problem, we need a key agreement protocol that also fixes long-term pairwise authentication keys so that we can re-run key agreement protocols without OOB communication. In particular, we can use public keys corresponding to Diffie-Hellman key agreement protocol. The corresponding key agreement protocol was proposed by Laur and Pasini (2008) and it is depicted in Figure 12.

As the transcript of the Burmester-Desmedt protocol is authenticated with the SAS-GMA, the protocol is immune against active attacks with the same guarantees as Theorems 6 and 9 specify. Moreover, any two parties $\alpha, \beta \in \mathcal{H}$ can establish a pairwise secret key $\mathsf{sk}_{\alpha,\beta} = f(g^{x_\alpha x_\beta})$, as they both know the corresponding long-term public keys $y_i = g^{x_i}$ for all group members $i \in \mathcal{G}$. Hence, they can use any classical authentication protocol to protect new instances of group key agreement protocols against active attacks. In particular, we can merge small groups $\mathcal{G}_1, \mathcal{G}_2$, if there is an honest party $\mathcal{P}_i \in \mathcal{G}_1 \cap \mathcal{G}_2$, by sending all intergroup communication through $\mathcal{P}_i$.

Of course, if the formed group is known to have a static nature, then one can skip the setup of long-term Diffie-Hellman keys $\mathsf{sk}_{\alpha,\beta}$.

8 Conclusions and open problems

As explained in the introduction, setting up a secure communication between two or more parties requires authenticated communication channels. As a solution to this problem, we presented several user-aided two-party message authentication protocols in Section 4. Note that three of these protocols are optimal which means that no protocol can achieve a better security using the same amount of authentic data. We conclude that we cannot

Figure 11 The Burmester-Desmedt group key agreement protocol

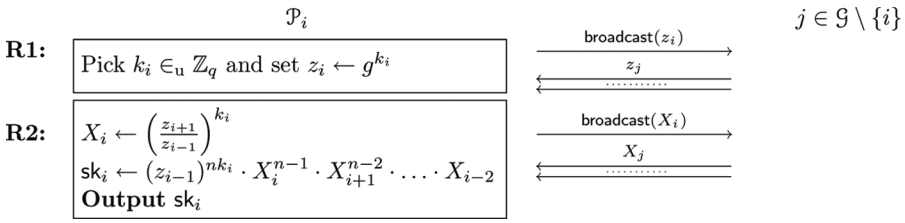
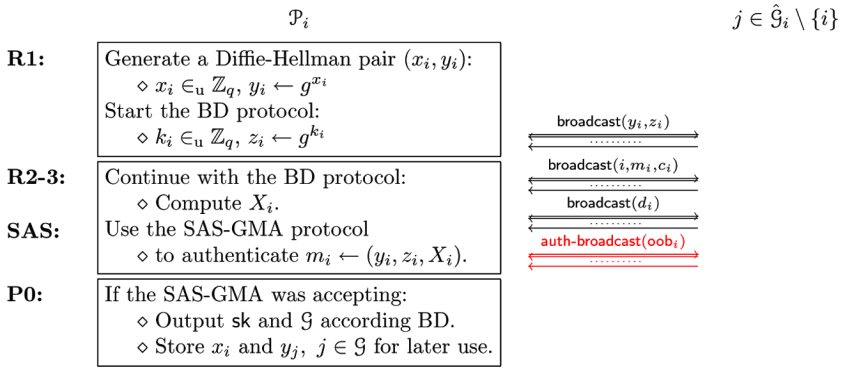


Figure 12 The final SAS-based AKA protocol with simplified notations (see online version for colours)



do much better in two-party settings. However in group settings, the use of two-party protocols is not optimal.

Indeed, suppose we want to setup a shared secret key for more than two parties. Using two-party protocols, we should run several peer-to-peer (two-party) protocols and this solution increases the amount of user interaction. In Section 5 we addressed this concern and presented a group message authentication protocol that is significantly better in that respect and achieves the optimal security level.

Another remarkable aspect in this paper is a well-adapted adversarial model (see Section 3) and a systematic treatment of various execution models (covered by Sections 4–6). More precisely, we first proved that all presented protocols are secure in the stand-alone model and then showed that any user-aided message authentication protocol that is secure in the stand-alone model remains secure in more complex settings.

As a last step in securing communication links, we presented a general methodology how to protect ordinary key agreement protocols against active attacks. Since the corresponding technique is well established for two-party case, we gave the detailed description only for the group setting. We emphasise here that the final protocol depicted in Figure 12 has an optimal security with respect to the amount of authentic data as well as an optimal number of rounds. Additionally, the clever use of long-term public keys provides an efficient way to manage dynamic groups.

To emphasise differences between various protocols, we have gathered the most important aspects into Table 1. For brevity, we use the acronym MA for unilateral message authentication, MCA for cross-authentication, GKA for group message authentication. To distinguish between unkeyed and keyed hash functions, we use shorthands $h(\cdot)$ and $h_K(\cdot)$. A tick indicates that the protocol has the property that is specified by the column.

Table 1 Main features of user-aided message authentication protocols

Type	*Protocol*	*Interactive*	*Optimal*	*Primitives*
MA	Fingerprint			$h(\cdot)$
	MANA I and II			$h_K(\cdot)$
	MANA III	✓		$h_K(\cdot)$
	SAS (original)	✓	✓	$\mathsf{com}(m)$
	Pasini-Vaudenay		✓	$\mathsf{com}(m), h(\cdot)$
MCA	SAS MCA	✓		$\mathsf{com}(m)$
	MANA IV	✓	✓	$\mathsf{com}(m), h_K(\cdot)$
	Optimal SAS-MCA	✓	✓	$\mathsf{com}(m), h_K(\cdot)$
GMA	G-MANA IV	✓	✓	$\mathsf{com}(m), h_K(\cdot)$
	SAS-GMA	✓	✓	$\mathsf{com}(m), h_K(\cdot)$

As an intricate theoretical detail, note that it is quite straightforward to prove that all user-aided authentication protocols with optimal security level must have at least three moves over insecure channel, see for example Laur and Nyberg (2006). However, the result does not specify the maximal security level for two-move protocols. Hence, it is theoretically interesting to know what is the corresponding lower bound on deception probability and whether all two-move protocols are as sub-optimal as the MANA II protocol.

There is also a tension between theoretical constructions and practical instantiations. Most practical user-aided key agreement protocols such as Zfone protocol (Zimmermann et al., 2000) and wireless USB key agreement protocol (WUS, 2006) use collision resistant hash functions to mimic the functionality of commitment scheme. Although this approach cannot be used in general, it should be appropriate for securing key agreement protocols, since the corresponding authenticated messages have uniform distribution.

More formally, a collision resistant hash function as a deterministic commitment cannot be hiding and non-malleable for arbitrary message distribution. However, hiding and non-malleability w.r.t. uniform distribution makes sense also for hash functions. Consequently, it should be possible to give a formal security proof for these practical protocols. On the other hand, the corresponding security requirements are very different from the standard ones such as one-wayness and collision resistance. Hence, interpretation of corresponding security requirements is an interesting theoretical and practical problem.

Acknowledgements

Partially supported by the Academy of Finland and by the Estonian Doctoral School in Information and Communication Technology. Supported by the Swiss National Science Foundation, 200021-113329.

References

Balfanz, D., Smetters, D.K., Stewart, P. and Wong, H.C. (2002) 'Talking to strangers: authentication in ad-hoc wireless networks', *Proceedings of NDSS '02: The Network and Distributed System Security Symposium*, San Diego, California, USA.

Bellare, M., Desai, A., Pointcheval, D. and Rogaway, P. (1998) 'Relations among notions of security for public-key encryption schemes', in Krawczyk, H. (Ed.): *CRYPTO*, Volume 1462 of Lecture Notes in Computer Science, Springer-Verlag, Santa Barbara, California, USA, pp.26–45.

Bellare, M. and Rogaway, P. (1993) 'Entity authentication and key distribution', in Stinson, D.R. (Ed.): *CRYPTO*, Volume 773 of Lecture Notes in Computer Science, Springer-Verlag, Santa Barbara, California, USA, pp.232–249.

Bellare, M. and Rogaway, P. (1995) 'Provably secure session key distribution: the three party case', *STOC*, ACM, Las Vegas, Nevada, USA, pp.57–66.

Bellare, M. and Sahai, A. (1999) 'Non-malleable encryption: equivalence between two notions, and an indistinguishability-based characterization', in Wiener, M.J. (Ed.): *CRYPTO*, Volume 1666 of Lecture Notes in Computer Science, Springer-Verlag, Santa Barbara, California, USA, pp.519–536.

Burmester, M. and Desmedt, Y. (2005) 'A secure and scalable group key exchange system', *Information Processiong Letter*, Vol. 94, No. 3, pp.137–143.

Canetti, R. (2000) *Universally Composable Security: A New Paradigm for Cryptographic Protocols*, Cryptology ePrint Archive, Report 2000/067. http://eprint.iacr.org/

Canetti, R. (2001) 'Universally composable security: a new paradigm for cryptographic protocols', *FOCS*, Las Vegas, Nevada, USA, pp.136–145.

Canetti, R. and Krawczyk, H. (2002) 'Universally composable notions of key exchange and secure channels', in Knudsen, L.R. (Ed.): *EUROCRYPT*, Volume 2332 of Lecture Notes in Computer Science, Springer-Verlag, Amsterdam, The Netherlands, pp.337–351.

Crescenzo, G.D., Ishai, Y. and Ostrovsky, R. (1998) 'Non-interactive and non-malleable commitment', *STOC*, Dallas, Texas, USA, pp.141–150.

Damgård, I. and Groth, J. (2003) 'Non-interactive and reusable non-malleable commitment schemes', *STOC*, ACM, San Diego, CA, USA, pp.426–437.

Desmedt, Y. and Burmester, M. (1994) 'A secure and efficient conference key distribution system (extended abstract)', in De Santis, A. (Ed.): *EUROCRYPT '94*, Volume 950 of Lecture Notes in Computer Science, Springer-Verlag, Perugia, Italy, pp.275–286.

Diffie, W. and Hellman, M.E. (1976) 'New directions in cryptography', *IEEE Transactions on Information Theory, IT*, Vol. 22, No. 6, pp.644–654.

Dolev, D., Dwork, C. and Naor, M. (1991) 'Non-malleable cryptography', *STOC*, New York, NY, USA, ACM Press, Vol. 91, pp.542–552.

ElGamal, T. (1985) 'A public key cryptosystem and a signature scheme based on discrete logarithms', *IEEE Transactions on Information Theory*, Vol. 31, No. 4, pp.469–472.

Fischlin, M. and Fischlin, R. (2000) 'Efficient non-malleable commitment schemes', in Bellare, M. (Ed.): *CRYPTO*, Volume 1880 of Lecture Notes in Computer Science, Springer-Verlag, Santa Barbara, California, USA, pp.413–431.

Gehrmann, C., Mitchell, C.J. and Nyberg, K. (2004) 'Manual authentication for wireless devices', *RSA Cryptobytes*, Vol. 7, No. 1, pp.29–37.

Just, M. and Vaudenay, S. (1996) 'Authenticated multi-party key agreement', in Kim, K. and Matsumoto, T. (Ed.): *ASIACRYPT '96*, Volume 1163 of Lecture Notes in Computer Science, Springer-Verlag, Kyongju, Korea, pp.36–49.

Laur, S. and Nyberg, K. (2006) 'Efficient mutual data authentication using manually authenticated strings', in Pointceval, D., Mu, Y. and Chen, K. (Eds.): *Cryptology and Network Security*, Volume 4301 of Lecture Notes in Computer Science, Springer-Verlag, Suzhou, China, pp.90–107.

Laur, S. and Pasini, S. (2008) 'SAS-based group authentication and key agreement protocols', in Cramer, R. (Ed.): *PKC 2008*, Volume 4939 of Lecture Notes in Computer Science, Springer-Verlag, Barcelona, Spain, pp.197–213.

Lindell, Y. (2003) 'General composition and universal composability in secure multi-party computation', *FOCS*, IEEE Computer Society, Cambridge, MA, USA, pp.394–403.

MacKenzie, P.D. and Yang, K. (2004) 'On simulation-sound trapdoor commitments', in Cachin, C. and Camenisch, J. (Eds.): *EUROCRYPT*, Volume 3027 of Lecture Notes in Computer Science, Springer-Verlag, Interlaken, Switzerland, pp.382–400.

Mashatan, A. and Stinson, D.R. (2006) *Noninteractive Two-Channel Message Authentication Based on Hybrid-Collision Resistant Hash Functions*, Cryptology ePrint Archive, Report 2006/302. http://eprint.iacr.org/

Maurer, U.M. (2000) 'Authentication theory and hypothesis testing', *IEEE Transactions on Information Theory*, Vol. 46, No. 4, pp.1350–1356.

Pasini, S. and Vaudenay, S. (2006a) 'An optimal non-interactive message authentication protocol', in Pointcheval, D. (Ed.): *CT-RSA*, Volume 3860 of Lecture Notes in Computer Science, Springer-Verlag, San Jose, CA, USA, pp.280–294.

Pasini, S. and Vaudenay, S. (2006b) 'SAS-based authenticated key agreement', in Yung, M., Dodis, Y., Kiayias, A. and Malkin, T. (Eds.): *PKC 2006*, Volume 3958 of Lecture Notes in Computer Science, Springer-Verlag, New York, NY, USA, pp.395–409.

Pfitzmann, B. and Waidner, M. (2001) 'A model for asynchronous reactive systems and its application to secure message transmission', *IEEE Symposium on Security and Privacy*, Oakland, California, USA, pp.184–200.

Reyhanitabar, M.R., Wang, S. and Safavi-Naini, R. (2007) 'Non-interactive manual channel message authentication based on etcr hash functions', in Pieprzyk, J., Ghodosi, H. and Dawson, E. (Eds.): *ACISP*, Volume 4586 of Lecture Notes in Computer Science, Springer-Verlag, Townsville, Australia, pp.385–399.

Rivest, R.L., Shamir, A. and Adleman, L.M. (1978) 'A method for obtaining digital signatures and public-key cryptosystems', *Communications of the ACM*, Vol. 21, No. 2, pp.120–126.

Simmons, G.J. (1984) 'Authentication theory/coding theory', in Blakley, G.R. and Chaum, D. (Eds.): *CRYPTO '84*, Volume 196 of Lecture Notes in Computer Science, Springer, Santa Barbara, California, USA, pp.411–431.

Valkonen, J., Asokan, N. and Nyberg, K. (2006) 'Ad hoc security associations for groups', in Buttyan, L., Gligor, V.D. and Westhoff, D. (Eds.): *ESAS*, Volume 4357 of Lecture Notes in Computer Science, Springer-Verlag, Cambridge, UK, pp.150–164.

Vaudenay, S. (2005) 'Secure communications over insecure channels based on short authenticated strings', in Shoup, V. (Ed.): *CRYPTO*, Volume 3621 of Lecture Notes in Computer Science, Springer-Verlag, Santa Barbara, California, USA, pp.309–326.

WUS (2006) *Association Models Supplement to the Certified Wireless Universal Serial Bus Specification*, http://www.usb.org/developers/wusb/...../wusb_2007_0214.zip

Zimmermann, P., Johnston, A. and Callas, J. (2000) *ZRTP: Media Path Key Agreement for Secure RTP*, http://www3.tools.ietf.org/html/ ... /draft-zimmermann-avt-zrtp-04.

Notes

[1] We always assume that the group $\mathcal{G}$ is ordered w.r.t. the sender identities $\mathsf{id}_1 < \mathsf{id}_2 < \cdots < \mathsf{id}_n$.

[2] Strictly speaking, such an assumption is unnecessary but it guarantees reusability of a common reference string.

[3] In fact, the Diffie-Hellman key agreement is known to satisfy this requirement modulo minor details (Canetti and Krawczyk, 2002).

Appendix

A Security of the SAS protocol

For the proof of Theorem 3, we have to show that the inequalities (1)–(3) defined below hold for any t-time adversary $\mathcal{A}$ whenever the assumptions of Theorem 3 are satisfied. Let forge denotes the event that $\mathcal{B}$ succeeds in deception. Then we can express

$$\mathrm{Adv}^{\mathsf{forge}}(\mathcal{B}) \le \Pr[\mathsf{forge} \wedge c \neq \hat{c}] + \varepsilon_{\mathrm{b}}, \tag{1}$$

since a successful forgery such that $c = \hat{c}$ reveals a double opening and thus $\Pr[\mathsf{forge} \wedge c = \hat{c}] \le \varepsilon_{\mathrm{b}}$. For further analysis, let $\hat{c} \prec d$ denote the event that $\mathcal{P}_2$ receives $\hat{c}$ before than $\mathcal{P}_1$ releases d. Then

$$\Pr[\mathsf{forge} \wedge c \neq \hat{c} \wedge \hat{c} \prec d] \le 2^{-\ell} \cdot \Pr[\hat{c} \prec d] + \varepsilon_{\mathrm{nm}} \tag{2}$$

or otherwise we can construct $t + \mathcal{O}(1)$-time adversary $\mathcal{A}$ that simulates the stand-alone model for $\mathcal{B}$ in order to win the corresponding non-malleability games.

Namely, the adversary $\mathcal{A}_1$ feeds pk to $\mathcal{B}$ and gets back corresponding input message m. Next, $\mathcal{A}_1$ defines MGen as a uniform distribution over pairs (m, r_1) where $r_1 \in \{0,1\}^{\ell}$ and uses the challenger's reply as c in the SAS protocol. To be precise, $\mathcal{A}_1$ faithfully simulates the SAS protocol to $\mathcal{B}$ and halts if $d \prec \hat{c}$. As an intermediate output $(\sigma, \hat{c}_1, \ldots, \hat{c}_n)$, the adversary $\mathcal{A}_1$ outputs $\sigma = (m, \hat{m}, \hat{r}_2, r_2)$ and $\hat{c}$. As a final output $(\hat{d}_1, \ldots, \hat{d}_n)$, $\mathcal{A}_1$ outputs the corresponding decommitment value $\hat{d}$. Now $\mathcal{A}_2(\sigma, x_i, \hat{y})$ resumes the SAS protocol: sets $(m, r_1) \leftarrow x_i$ and $(\hat{m}, \hat{r}_1) \leftarrow \hat{y}$ and output 1 only if $m \neq \hat{m}$ and $r_1 = \hat{r}_1$. Now by construction $\Pr[\mathcal{G}_0^{\mathsf{nm}} = 1] = \Pr[\mathsf{forge} \wedge c \neq \hat{c} \wedge \hat{c} \prec d]$ and $\Pr[\mathcal{G}_1^{\mathsf{nm}} = 1] \le 2^{-\ell} \cdot \Pr[\hat{c} \prec d]$, since x_1 is independent of the protocol run and thus $\Pr[r_1 = \hat{r}_1 | \mathcal{A}_1 \neq \perp] = 2^{-\ell}$.

As a final detail, we have to prove the inequality

$$\Pr[\mathsf{forge} \wedge c \neq \hat{c} \wedge d \prec \hat{c}] \le 2^{-\ell} \cdot \Pr[d \prec \hat{c}] + \sqrt{\varepsilon_{\mathrm{b}}}. \tag{3}$$

The corresponding proof is technically tedious, as the adversary does not directly violate the binding property. Consequently, we have to construct a collision-extractor $\mathcal{A}$ that runs in time $2t + \mathcal{O}(1)$ and finds the double opening with high enough probability. We give here only the proof sketch, since analogous proof that covers all details can be found in Laur and Nyberg (2006). Consider an adversary $\mathcal{A}$ that simulates the protocol to $\mathcal{B}$ with two independent r_2 values r_2^0 and r_2^1 to get two protocol transcripts with the same commitment value $\hat{c}$ but different decommitment values $\hat{d}_0$ and $\hat{d}_1$.

That is, $\mathcal{A}$ first simulates the protocol with r_2^0 and then rewinds $\mathcal{B}$ and submits r_2^1 and outputs the corresponding triple $(\hat{c}, \hat{d}_0, \hat{d}_1)$. Let succ denote the event that $\mathcal{A}$ gets a double opening. Then we can lower bound the success probability $\Pr[\mathsf{succ} | d \prec \hat{c}]$ by the conditional averages

$$\mathbf{E}_{\hat{c}}(\Pr[\mathsf{forge}|\hat{c}]^2 \,|\, d \prec \hat{c}) - \mathbf{E}_{\hat{c}}(\Pr[\mathsf{forge}|\hat{c}] \cdot 2^{-\ell} | d \prec \hat{c}).$$

Hence, the Jensen inequality $\mathbf{E}(X^2) \ge \mathbf{E}(X)^2$ assures that

$$\Pr[\mathsf{succ} \mid d \prec \hat{c}] \ge \Pr[\mathsf{forge} \mid d \prec \hat{c}]^2 - 2^{-\ell} \cdot \Pr[\mathsf{forge} \mid d \prec \hat{c}].$$

Now if the inequality (3) is violated, then we can also conclude that $\Pr[\mathsf{forge} | d \prec \hat{c}] \ge 2^{-\ell}$. Consequently,

$$\Pr[\mathsf{succ} | d \prec \hat{c}] \ge (\Pr[\mathsf{forge} | d \prec \hat{c}] - 2^{-\ell})^2$$

and we have derived a contradiction

$$\Pr[\mathsf{succ}] \ge \Pr[d \prec \hat{c}]^2 \cdot (\Pr[\mathsf{forge} | d \prec \hat{c}] - 2^{-\ell})^2 \ge \varepsilon_{\mathrm{b}}.$$

B Security of the optimised SAS-MCA protocol

For the proof of Theorem 4, we show that the inequalities (4)–(6) defined below hold for any t-time adversary whenever the assumptions of Theorem 4 are satisfied. Again, we assume that $\mathcal{B}$ is a t-time adversary. First, note that the proof of the inequality

$$\Pr[\mathsf{forge} \wedge d \prec \hat{c}] \le 2^{-\ell} \cdot \Pr[d \prec \hat{c}] + \sqrt{\varepsilon_{\mathrm{b}}}, \tag{4}$$

is analogous to Appendix A. Secondly, note that the case $c = \hat{c}$ requires more detailed analysis, as adversary can also alter the second message m_2. In fact, the corresponding upper bound is also more complex

$$\Pr[\mathsf{forge} \wedge c \neq \hat{c} \wedge \hat{c} \prec d] \le \varepsilon_{\mathrm{u}} \cdot \Pr[m_1 = \hat{m}_1 \wedge c = \hat{c} \wedge \hat{c} \prec d] + \varepsilon_{\mathrm{b}} + \varepsilon_{\mathrm{h}}. \tag{5}$$

For the proof, note that c can be opened to $(\hat{m}_1, \hat{r}_1)$ that is different from (m_1, r_1) with probability at most ε_{b}. Now consider a $t + \mathcal{O}(1)$-time adversary $\mathcal{A}$ that simulates the protocol to $\mathcal{B}$ in order to win the hiding game. Namely, given message m, $\mathcal{A}$ submits two messages $x_0 = (m, r_1)$ and $x_1 = (m, r_1^*)$ where $r_1, r_1^* \in_{\mathrm{u}} \{0,1\}^s$ to the challenger, then uses the challenge commitment c_s in the simulation and then stops when we need the decommitment d. Next, $\mathcal{A}$ halts if $d \prec \hat{c}$ or $c \neq \hat{c}$. Otherwise, $\mathcal{A}$ computes $\mathsf{oob}_1 \leftarrow h(\hat{m}_2, r_1) \oplus \hat{r}_1$ and $\mathsf{oob}_2 \leftarrow h(m_2, r_1) \oplus r_2$. $\mathcal{A}$ outputs 0 if B succeeds in deception, i.e., $\mathsf{oob}_1 = \mathsf{oob}_2$ and $m_2 \neq \hat{m}_2$ and 1 otherwise. Now it is straightforward to verify that if the inequality (5) is violated then $\mathrm{Adv}^{\mathsf{hid}}(\mathcal{A}) > \varepsilon_{\mathrm{h}}$.

Finally, note that non-malleability property assures

$$\Pr[\mathsf{forge} \wedge c \neq \hat{c} \wedge \hat{c} \prec d] \le \varepsilon_{\mathrm{r}} \cdot \Pr[c \neq \hat{c} \wedge \hat{c} \prec d] + \varepsilon_{\mathrm{nm}}, \tag{6}$$

or otherwise we can construct a $t + \mathcal{O}(1)$-time adversary $\mathcal{A}$ for the non-malleability games. The adversary $\mathcal{A}_1$ simulates the protocol execution analogously to the one described in Appendix A. The target relation $\mathcal{A}_2(\sigma, x_i, \hat{y})$ still outputs 1 only in the case of successful deception, i.e., $(m_1, \hat{m}_2) \neq (\hat{m}_1, m_2)$ and $h(\hat{m}_2, r_1) \oplus \hat{r}_2 = h(m_2, \hat{r}_1) \oplus r_2$. The only difference in the argumentation comes from the fact that $\mathcal{A}$ can win the $\mathcal{G}_1^{\mathsf{nm}}$ with probability at most $\Pr[\mathcal{G}_1^{\mathsf{nm}} = 1 | \mathcal{A}_1 \neq \perp] \leq \varepsilon_{\mathrm{r}}$, since r_1 is chosen independently of all other values needed to compute OOB messages. Secondly, $\Pr[\mathcal{A}_1 \neq \perp] = \Pr[c \neq \hat{c} \wedge \hat{c} \prec d]$ and thus $\mathrm{Adv}^{\mathsf{nm}}(\mathcal{A}) > \varepsilon_{\mathrm{nm}}$ as soon as the inequality (6) violated.