



ÉCOLE POLYTECHNIQUE  
FÉDÉRALE DE LAUSANNE

---

# Secure Communications over Insecure Channels Using an Authenticated Channel

Sylvain Pasini  
sylvain.pasini@epfl.ch

Master Thesis Work  
School of Computer and Communication Sciences

6<sup>th</sup> September 2005

Professor Serge Vaudenay  
serge.vaudenay@epfl.ch  
EPFL / LASEC





To Elysa, my goddaughter



# Acknowledgements

I would like to express my gratitude to my supervisor, Professor Serge Vaudenay, who offered me the chance to carry out my master thesis at the LASEC. Everything started in March 2004 with my first course “Security and Cryptography”. I would like to start thanking Serge Vaudenay for having succeeded to pass his knowledge of cryptography to me. I was about to discover a new passion. Indeed, Cryptography is a mysterious world. I always have found funny the way mathematical tools can be applied to practical systems. A cryptograph has a special way thinking: everytime, everywhere, he tries to find “holes in every wall”. Since then, I carried out a semester project and the present master thesis in LASEC. Thanks to his experience, vast knowledge, and rigour, I had the opportunity to discover the world of research. With his assistance, I was able to submit my first paper and to write the document you are holding.

During the six months of my master thesis, I was pleased to collaborate with all members of LASEC. The environment within this laboratory was very friendly and gave me the opportunity to acquire knowledge in many domains. I believe that all members deserve my thanks<sup>1</sup>:

As usual, Gildas Avoine is the first cited. I remember very good moments spent with him. I thank him for that and for all his advice, especially in  $\text{\LaTeX}$ . Between motorcyclists, the current is always on...

The person who supervised me during my semester project, and advised me to carry out my master thesis at the LASEC, is Thomas Baignères. Without him, I would not be part of the crypto world today. Thanks Thomas!

Let us not forget the two distant PhD students from Gemplus, Claude Barral and Julien Brouchier, who came a few times to visit us and who brought us the sun from the south. Unfortunately, they always took it back with them.

Our secretary Matine Corval deserves to be thanked, especially for her availability when it comes to helping us including administrative tasks (to which I really don't understand anything).

I would especially like to thank Matthieu Finiasz for all his help. Thanks to his very good availability, his friendliness, his patience, and his knowledge, I had many opportunities to exchange ideas and to expose problems, often coming out of his office with a solution. I get along very well with him and he became one of my best friends.

---

<sup>1</sup>alphabetic order

Yi Lu was the only person sharing my room and I thank her for keeping me company.

Jean Monnerat, always of good mood and ready to defend Switzerland or his canton (the beautiful Jura), gave place to many animated debates. I thank him for his help, especially in mathematics and  $\text{\LaTeX}$ .

The newest member of the LASEC, Martin Vuagnoux, deserves to be thanked for his funny stories which make us laugh so much, especially his spectacular side-channel attacks.

Sharing the majority of our projects, spending much hours together (mostly in the train), Israel Hernandez became one of my best friends. I also thank him for his assistance during this work, including his help in Java programming.

Dr. Michel Kocher, my supervisor during my first diploma at the Engineering School of Geneva, who motivated, and encouraged me also deserves my thanks. I would like to thank him for his collaboration during my months of diploma work and for advising me to continue my formation at the EPFL. Without him, I would never have known the EPFL.

My parents should obviously not be forgotten. Thanks to them, I had the chance (amongst other things!) to follow an excellent formation and I greatly thank them. My final thanks go to Nadia. The two of us shared both moments of happiness and a few more difficult times. It is probably these latter which helped bring us together even more. No matter when and why, she always kept encouraging me and took it upon herself to leave me enough spare time to conclude this work (even if it sometimes required a few negotiations!). I finally thank her for her Love to which I had never found any limit. I think now it would be difficult for me to live without her... I want finally to thank all three of them for their support during these few years of studies.

After this pleasant period, I wanted to continue and I thus postulated for a PhD thesis... Thanks to everyone!

# Abstract

A secure peer-to-peer communication over an insecure channel without any prior exchanged key can be established with the help of an authentication step to exchange a public key. Then, standard methods of public-key cryptography such as RSA can be used to communicate securely.

In this work, we concentrate on message authentication protocols which require an extra authenticated channel. We start by describing some possible human communications channels, such as telephone, and by analyzing them according to some properties, in particular authentication properties. Then, we recall some message authentication protocols which use an authenticated extra channel. In addition, we recall different types of authentication. For instance, we recall biometrics-based systems which use the ability of humans to recognize the voice of the distant user. We recall also distance bounding-based systems which assumes that there is no other systems in the “integrity area”.

In a second step, we prove the maximal security of a message authentication protocol against adversaries and we show that a protocol using  $k$ -bit authenticated strings reaches the maximal security when the distribution among all possible authenticated strings is uniform. More precisely, we sketch three generic attacks against any message authentication protocol. Using these results, we study the security of different authentication protocols, either non-interactive or interactive.

In addition, we propose a new protocol which achieves the same security level against offline attacks as that of the one used today in many systems, such as SSH or GPG, but using much less authenticated bits.

Finally, we compare interactive and non-interactive authentication protocols and we study their usability in different applications.



# Contents

<b>Introduction</b>	<b>11</b>
<b>1 The Authentication Problem</b>	<b>13</b>
1.1 The Human Communication Channels . . . . .	13
1.2 Setup a Secure Communication over an Insecure Channel . . . . .	15
1.2.1 Conventional Cryptography . . . . .	15
1.2.2 Setting up Secure Channels without Confidential Channel . . . . .	16
1.2.3 Public-Key Cryptography . . . . .	16
<b>2 Message Authentication Nowadays</b>	<b>19</b>
2.1 Protocols Using an Authenticated Channel . . . . .	19
2.2 Protocols Using Biometric Signals . . . . .	21
2.3 Protocols Using Distance Bounding . . . . .	23
<b>3 Preliminaries</b>	<b>27</b>
3.1 Adversarial Model . . . . .	27
3.2 Authenticated Channels . . . . .	29
3.2.1 Weak Authenticated Channels . . . . .	29
3.2.2 Stronger Authenticated Channels . . . . .	29
3.2.3 Examples . . . . .	30
3.3 Message Authentication Protocols . . . . .	30
3.4 Commitment Schemes . . . . .	31
3.4.1 Tag-Based Commitment Model . . . . .	32
3.4.2 Completeness, Hiding, and Binding Properties . . . . .	32
3.4.3 Ideal Commitment Model . . . . .	34
3.4.4 Extractable and Equivocable Commitment Schemes . . . . .	35
3.4.5 Trapdoor Commitment Model . . . . .	36
3.4.6 Examples . . . . .	37
3.5 Hash functions . . . . .	40
3.5.1 Collision-Resistant Hash Functions . . . . .	40
3.5.2 Weakly Collision-Resistant Hash Functions . . . . .	40
3.5.3 Universal Hash Functions Families . . . . .	40

<b>4</b>	<b>On the Required Entropy of Authenticated Communications</b>	<b>41</b>
4.1	A Generic One-Shot Attack . . . . .	42
4.2	A Generic Multi-Shot Attack . . . . .	43
4.3	A Generic Multi-Shot Attack Against Non-Interactive Protocols . . . . .	44
4.4	A Short Overview . . . . .	44
<b>5</b>	<b>Non-Interactive Authentication Protocols</b>	<b>45</b>
5.1	A Non-Interactive Authentication Protocol Based on a Collision-Resistant Hash Function . . . . .	46
5.2	A Non-Interactive Authentication Protocol using Strong Authentication . . .	46
5.3	A Proposed Non-Interactive Authentication Protocol using Weak Authentication Only . . . . .	48
<b>6</b>	<b>Interactive Authentication Protocol</b>	<b>53</b>
6.1	The Vaudenay SAS-based Protocol . . . . .	53
6.2	Short Security Analysis, Optimality . . . . .	54
6.3	A First Application: OpenSSH . . . . .	54
6.4	A Peer-To-Peer File Authentication . . . . .	57
6.4.1	The Implemented Protocol . . . . .	57
6.4.2	The Final Application . . . . .	59
6.4.3	Number of SAS Trials . . . . .	60
6.4.4	In Short . . . . .	60
<b>7</b>	<b>Interactive vs. Non-Interactive Protocols and their Applications</b>	<b>63</b>
7.1	A Short Comparison . . . . .	63
7.2	Some Applications Examples . . . . .	64
7.2.1	Applications Based on Public Keys . . . . .	64
7.2.2	Bluetooth Devices Pairing and Wireless USB . . . . .	65
	<b>Conclusion</b>	<b>67</b>
	<b>Bibliography</b>	<b>69</b>
	<b>List of Figures</b>	<b>75</b>
	<b>Appendix A</b>	<b>77</b>
	<b>Appendix B</b>	<b>79</b>

# Introduction

One key issue in cryptography is to achieve secure peer-to-peer communications over insecure channels. Chapter 1 describes how to establish such a channel. In short, using conventional cryptography, like DES [DES77], it is necessary that the users share a confidential key  $K$ . The only human manner to exchange a private key is to encounter. Phone, mail and email are not secure. Consequently, the users must meet face to face which can be hard and expensive. Using the Diffie-Hellman model [DH76] the key establishment has been reduced to the exchange of authenticated messages. Using public key cryptography, like RSA [RSA78], we know that it is possible to reduce the problem of establishing a secure communication over an insecure channel by exchanging public keys in an authenticated way. This work is about how this exchange is performed.

Different solutions were proposed to authenticate a public key. In Chapter 2 we recall some of them. In particular, we describe authentication protocol using distance bounding which is well adapted for devices pairing. We also describe a biometric-based system which use the ability of human beings to identify a voice. Finally, we recall three systems which use an extra authenticated channel such as telephone. These three are described in short below.

The most straightforward way to authenticate public keys is a scheme which was formalized by Balfanz et al. [BSSW02]. The proposed protocol is non-interactive and based on a collision resistant hash function. It simply consists of first sending the message and then authenticating its hashed value. The authenticated value must contains at least 160 bits (to be collision resistant) which is quite long. Note that this protocol is actually used in OpenSSH for the server public key authentication, in PGP and GPG for the exchange of personal public key, and many others.

Gehrmann-Mitchell-Nyberg have proposed in [GMN04] the MANA I Protocol. It allows to authenticate a message in a non-interactive way using an authenticated string of 16-20 bits only. On the other hand, the security requires strong assumptions on the authentication channel since the protocol is known to be insecure in other cases. Note that MANA protocols have been designed for wireless devices such as Bluetooth devices.

Vaudenay proposed in [Vau05] an interactive message authentication protocol based on Short Authenticated String (SAS). It is shown that a 15-bit SAS provides pretty good security which is much shorter than the first protocol presented. Note that it uses a weak authentication channel while MANA requires a stronger one.

The problem of message authentication using extra authenticated channels was also formalized by Vaudenay [Vau05]. The present thesis is based on this formalism. Chapter 3

describes the necessary preliminaries based on this formalism. First, this chapter starts by describing the adversarial model and defining authenticated channels and message authentication protocols. Then, it recalls some properties of commitment schemes and trapdoor commitment schemes. In particular, it explains that a commitment scheme can be seen as a “locked combination safe” which allows a user to commit on a value without revealing it. Finally, it defines properties of hash functions.

In Chapter 4, we analyze the maximal security of a general message authentication protocol. First, we consider only one-shot adversaries which can use one instance of each two peers. Then, we generalize to powerful adversaries that can use several instances of the two peers, i.e. considering collision attacks. In addition, we prove that a message authentication protocol is optimal when it uses uniformly distributed authenticated strings.

In Chapters 5 and 6, we discuss the security of the above message authentication protocols. We propose a new non-interactive protocol based on a commitment scheme and weakly collision resistant hash function. Our protocol allows shorter authenticated strings than the ones required by the protocol presented by Balfanz et al. [BSSW02]. We prove that our proposed protocol is optimal and we deduce the non-optimality of the other protocol.

The Vaudenay SAS-based protocol [Vau05] had never been used in a practical implementation. In Section 6.3, we propose an implementation in OpenSSH which is a straightforward application. A first advantage of [Vau05] is the fact that users are forced to enter the SAS since in the other case they can not establish the connection. More precisely, users can not agree the key without authentication. A second advantage is the short length, in particular when users must establish a secure communication quickly. Suppose for example a disaster cause in a bank and the administrator is abroad. Thus, he must establish a very quick connection with the server using a new generated pair of keys. The administrator is forced to authenticate the public key quickly. Using the SAS-based protocol, he has just to call the bank, starts the protocol, and finally types the SAS which has been obtained with the help of a responsible person.

In this application, the interactivity of the protocol is a problem: Suppose a client would open a connection with an OpenSSH server using the SAS-based protocol. He requires to enter the authenticated SAS which is in general transmitted by telephone. Consequently, the administrator must reply to telephone and transmit the SAS for each new connection in his park of hosts. Clearly, the administrator becomes quickly overloaded by the fact of having to transmit many SAS. We have thus to discuss about interactivity and non-interactivity as it is made in Chapter 7.

Finally, we propose an application of the SAS-based authentication protocol proposed by Vaudenay [Vau05] in which the interactivity is not a problem. In Section 6.4, we consider file authentication and in particular peer-to-peer file authentication. Note that a public key is in reality a file and thus can be authenticated by using this new application. In this example, the SAS can be exchanged by telephone between the two persons as if they had to exchange a fingerprint but exchanging only 15 bits instead of the 160 bits of a fingerprint.

# The Authentication Problem

## 1.1 The Human Communication Channels

Humans can communicate in different ways, i.e. using different human communication channels. Depending on the requirements, he chooses one of them. For example, if a human being needs to reach another human being for urgent matters, he must choose a channel with high availability and low latency such as a telephone link. But, if he would transfer an amount of money, he must establish a reliable authentication by going to the desk to encounter the person of the bank. (Nowadays, he can use the Internet with prior established security association.)

The security of communication channels can be characterized by some *security attributes* which are defined below.

**Definition 1** (Security Attributes). *Suppose a communication channel between a sender, called Alice, and a receiver, called Bob. A message  $m$  is sent on the input and a message  $\hat{m}$  can be read on the output. We define the following security properties:*

**Confidentiality** *assumes that only the legitimate receiver, i.e. Bob, can read the message  $\hat{m}$ .*

**Integrity** *assumes that the received message  $\hat{m}$  is the same as the input message  $m$ , i.e.  $\hat{m} = m$ .*

**Authenticity** *assumes that only the legitimate sender, i.e. Alice, can input a message  $m$  into the channel. This is often combined with integrity, i.e.  $m = \hat{m}$  can only be issued by Alice.*

**Freshness** *assumes that the received message  $\hat{m}$  was not received before.*

**Liveliness** *assumes that a message  $m$  which has been sent by Alice will eventually be delivered to Bob.*

**Timeliness** *assumes that a message  $m$  which has been send by Alice will be delivered immediately to Bob.*

In addition, to compare the different human communication channels, it is necessary to define other properties which characterize the *usability* of these channels. These *communication properties* are defined below.

**Definition 2** (Communication Properties). *Suppose a communication channel between a sender, called Alice, and a receiver, called Bob. We define the following communication properties:*

**The cost** *represents the required amount of money spent to establish the communication channel and to transmit a message from Alice to Bob.*

**The availability** *expresses the fact that the channel can easily be established at any time.*

**The speed rate** *represents the amount of data that can be transfered from Alice to Bob for a fixed time duration.*

**The latency** *represents the amount of time between the moment when Alice sends the message and the moment when Bob receives it.*

Using Definition 1 and Definition 2, it is possible to compare the common human communication channels in a cryptographic way.

**Face to face conversation** allows perfect authentication, perfect integrity and in certain cases, confidentiality. In addition, freshness, liveliness, and timeliness are trivially ensured. However, this channel can have a very high cost if, for example, the two persons are far from each other. For the same reasons, the availability is also bad. Note that the communication has no latency but a low speed rate. In conclusion, this human channel achieves high security but low throughput.

**Telephone** is like a face to face conversation but allows a third party to spy the communication. Thus, this channel does not guarantee confidentiality. On the other hand, it has a much lower cost and a higher availability. In short, it guarantees authentication assuming that both users can recognize the remote voice.

**Mail**, like a postcard or a parcel, is not confidential either. It can be easily lost and thus this channel does not guarantee liveliness. We can consider that a handwritten mail achieves authentication by assuming that the recipient can identify the writing. As for telephone, this channel guarantees availability but has a long latency.

**Electronic mail** is the worst communication channel in terms of security, it protects nothing by itself. However, it is the easiest communication channel and its costs is very small (too small if we consider the spam phenomenon), the availability and the speed rate are very high.

A short overview of the security and communication properties for each human communication channel is described on Fig. 1.1. Note that it is in fact a trade off between the security and the human usability. I.e. the more secure is a face to face conversation but it is the less

	Interactive		Non-interactive	
	Encounter	Telephone	Mail	Email
Authenticity	✓	✓	✓	
Integrity	✓	✓	✓	
Confidentiality	✓			
Freshness	✓	✓	✓	
Liveliness	✓	✓		
Timeliness	✓			
Cost		✓	✓	✓
Availability			✓	✓
Speed rate				✓
Latency	✓	✓		✓

**Figure 1.1.** The Common Human Communications Channels

usable. In the other hand, the more usable is the email channel but it is the less secure. For a specific use, we have to choose the better human communication channel depending on the security required and the available cost.

**Remark 1.** *Confidentiality is achieved only using a face to face conversation which can be very expensive in certain cases. However, a phone call, which is very easy to setup world-wide, achieves authentication assuming that the two speakers can identify the distant one by recognizing his voice.*

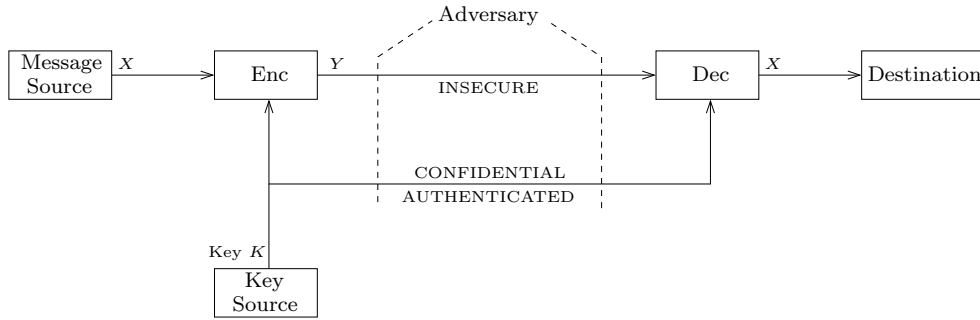
## 1.2 Setup a Secure Communication over an Insecure Channel

One of the main cryptographic issues is the problem of establishing a secure peer-to-peer communication over an insecure channel. Often, users can use an extra channel which achieves confidentiality and/or authenticity. In this section, some solutions are analyzed depending on the extra channel assumptions.

### 1.2.1 Conventional Cryptography

Using conventional cryptography, it is possible to establish a secure peer-to-peer channel assuming that we can establish a private and authenticated key. In fact, we use the Shannon Model [Sha49] which is depicted on Fig. 1.2. Using this model, confidentiality can be achieved using symmetric encryption. Symmetric encryption can be done using either stream ciphers, like E0 [Blu03], or block ciphers, like DES [DES77], AES [AES01], or FOX [JV03]. Authenticity and integrity can also be achieved using message authentication codes (MAC). In general, MAC can be constructed from stream ciphers or block ciphers, like the One-Key CBC MAC (OMAC) by Iwata and Kurosawa [IK03], or hash functions, like HMAC by Bellare-Cannetti-Krawczyk [BCK96]. Note that confidentiality, authenticity and integrity can be achieved at the same time using a combined mode.

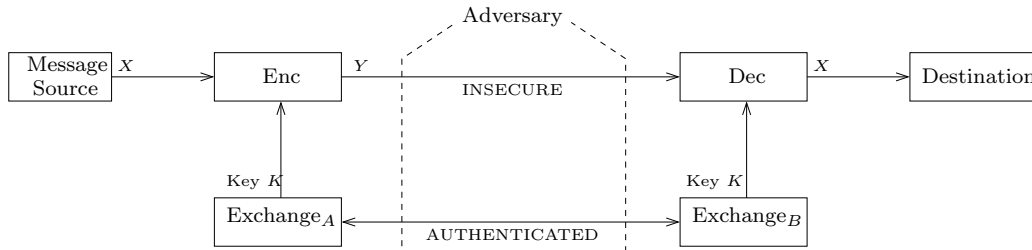
From the previous section, one can note that a face to face conversation is the only common human communication channel which achieves confidentiality, see Remark 1. These human communication channels can be very expensive and hard to establish. One would therefore favor the usage of channels which do not require confidentiality.



**Figure 1.2.** The Shannon Model [Sha49].

### 1.2.2 Setting up Secure Channels without Confidential Channel

In Merkle [Mer78] and Diffie-Hellman [DH76], it was shown a way to reduce the confidential extra channel to an authenticated extra channel. This channel is used to agree on a private key only by authenticating the exchanged messages. This is described by the Merkle-Diffie-Hellman model [Mer78, DH76] which is depicted on Fig. 1.3.

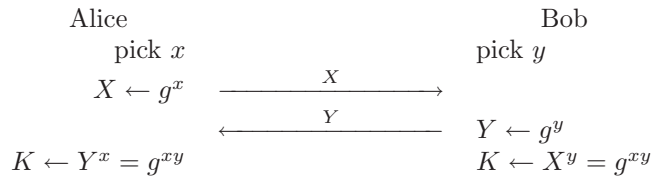


**Figure 1.3.** The Merkle-Diffie-Hellman Model.

An example is the Diffie-Hellman key agreement protocol [DH76] which is depicted on Fig. 1.4. Each party knows the public parameter  $g$  which spans a group  $G$ . Alice, resp. Bob, picks a random number  $x$ , resp.  $y$  and computes  $X \leftarrow g^x$ , resp.  $Y \leftarrow g^y$ . Then, Alice sends  $X$  to Bob and Bob sends  $Y$  to Alice. Alice, resp. Bob, computes  $Y^x$ , resp.  $X^y$ , which are in fact  $g^{xy}$ . Thus, they share a secret key  $K = g^{xy}$ . Note that  $g$  is chosen such that for any adversary who knows  $X$  and  $Y$  it is hard to retrieve  $x$  and  $y$  (Discrete Logarithm Problem). Consequently, it is hard to find the key  $K$ . On the other hand, without authentication an adversary can run a man-in-the-middle attack between the two participants, so authentication is required for this protocol.

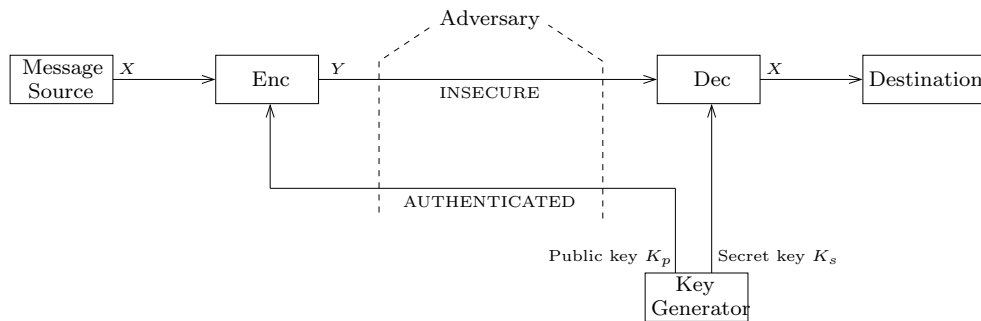
### 1.2.3 Public-Key Cryptography

Using public-key cryptography, it is also possible to relax the private hypothesis (confidentiality) on the extra channel. It can further be assumed to be unidirectional. We obtain a semi-authenticated key transfer, for example using RSA [RSA78], as depicted in Fig. 1.5. For instance, it can be used to transfer a private symmetric key and thus establish a secure communication using an extra channel which achieves authenticity only. Thus, the required



**Figure 1.4.** The Diffie-Hellman Key Agreement Protocol.

channel, which before required confidentiality, has been reduced to an authenticated channel.



**Figure 1.5.** The Semi-Authenticated Key Transfer Using Public-Key Cryptography.

Remark 1 says that establishing a private channel, i.e. which achieves confidentiality and authenticity, is hard since it requires a face to face conversation. In addition, it says that an extra channel which achieves only authenticity can be established simply by telephone. Thus, we can setup a secure communication over an insecure channel without requiring to encounter, but simply using telephone by exchanging a public key in an authenticated way.

To conclude, establishing a secure peer-to-peer communication can be reduced to “simply” exchanging a public key in an authenticated way as described on Fig. 1.5. Note that RSA [RSA78] requires actually 1024 bits which is still long for authenticated human communication channels like telephone. We have thus to reduce the amount of authenticated bits by using protocols and by assuming that a fixed security  $S$  is enough, where  $S$  is a set containing the maximal probability of success  $p$ , the maximal complexity  $T$ , the maximal number of trials  $Q$  (instances). For instance, we can consider  $T \leq 2^{70}$ ,  $Q \leq 2^{10}$ , and  $p \leq 2^{-20}$ .



# Message Authentication Nowadays

In the previous chapter, we have reduced the problem of establishing a secure communication over an insecure channel to the problem of exchanging a public key in an authenticated way, e.g. using the telephone. We have also seen that a key is in general 1024 bits long.

In addition, we have seen some human authenticated channels that can be used. For instance, using telephone as the human authentication channel, it can take long time to authenticate the 1024 bits as is. Thus, human beings would use more *user-friendly* systems. In this chapter, we present protocols which allow to reduce the amount of authenticated data.

Many authentication methods exist and we describe briefly some of them. First, we discuss authentication methods which use an extra authenticated channel, such as telephone or mail (see Section 1.1). Then, we discuss about biometrics-based authentication. In particular, we talk about voice recorded signal that can be exchanged over the insecure channel. This records can be seen as an authenticated channel with particular assumptions. Finally, we discuss about distance bounding authentication. These methods check whether the two devices are alone in a bounded area, called *integrity area*, by measuring round trip times. Then, they conclude on an authenticated channel between them since they are ensured that they were communicating with no third party.

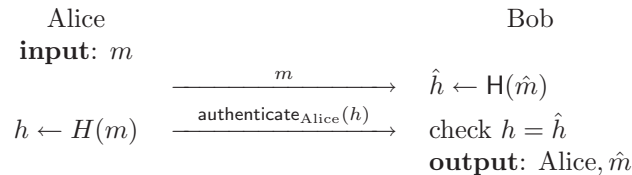
## 2.1 Protocols Using an Authenticated Channel

In this section, we present some message authentication protocols which can be used to authenticate a public key  $K_p$  by transmitting shorter authenticated strings through an authenticated channel and the complete message through an insecure channel.

Assume that Bob wants to authenticate the public key  $K_p$  of Alice with no prior exchanged keys. She can communicate over an insecure channel and over an extra authenticated channel. Note that the authenticated channel is expensive and she has to minimize the amount of authenticated data through them. Assume that the exchange of the public key (without

authentication) is easy, it can be done for example by email. Many solutions exist for Bob to authenticate the public key of Alice. In the following, we discuss about three message authentication protocols and we consider the more general problem of authenticating an arbitrary message  $m$ , instead of a public key  $K_p$  (e.g.  $m = K_p$ ).

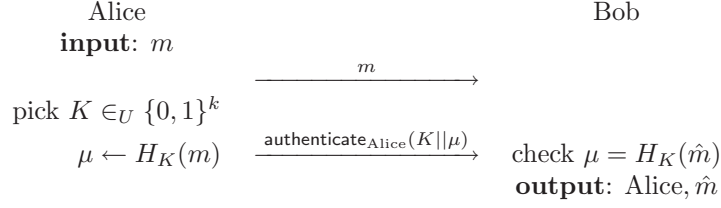
Today, the most common way to authenticate a message was presented by Balfanz et al. [BSSW02]. The protocol is based on a collision resistant hash function. It is depicted on Fig. 2.1. The sender of the message  $m$ , Alice, sends it to the recipient, Bob, using the insecure channel. Recall that  $m$  is not private. Note that we put a hat on the non authenticated values since they can differ from sent messages. Alice, resp. Bob, computes the hash value  $h$ , resp.  $\hat{h}$ , of the message  $m$ , resp. of the received message  $\hat{m}$ . Now, Alice has to authenticate the hashed value  $h$  to Bob. Comparing the received value  $h$  and his computed value  $\hat{h}$ , Bob can check whether the received message  $\hat{m}$  has been sent by the owner. One advantage of this protocol is that it is *non-interactive*. On the other hand, due to the non-interactivity, offline attacks are possible and are often very dangerous. Collision attacks work with a complexity of  $\mathcal{O}(2^{k/2})$  where  $k$  is the size of  $h$ . Thus, the authenticated message must contain at least 160 bits to avoid attacks using the birthday paradox.



**Figure 2.1.** Non-Interactive Message Authentication Using a Collision Resistant Hash Function.

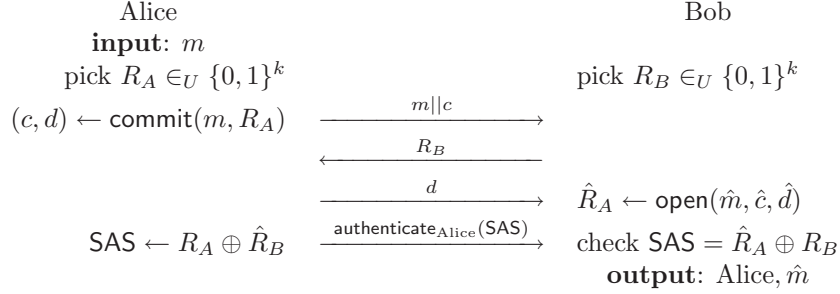
To reduce the size of the authenticated strings, Gehrman-Mitchell-Nyberg [GMN04] have proposed the MANA I protocol which is depicted on Fig. 2.2. As in [BSSW02], the sender of the message  $m$  Alice sends it to the recipient Bob using the insecure channel. As before, this step can be done for example by email. Then, she picks a key  $K$  uniformly in  $\{0, 1\}^k$  and computes  $\mu \leftarrow H_K(m)$ . Finally, she authenticates  $K||\mu$  to Bob. Bob can also check whether the hashed value of the received message  $\hat{m}$ , i.e.  $H_K(\hat{m})$ , is equal to  $\mu$ . The security requires a “stronger” authenticated channel since the protocol is known to be insecure with “weak” authentication only. Note that the randomness of the key  $K$  avoids collision attacks. Indeed, an adversary must run several instances of the protocol which is easy to detect. It allows to authenticate a message in a non-interactive way using only a 16-20 bits authenticated string.

Finally, an *interactive* message authentication protocol based on Short Authenticated String (SAS) which uses very few authenticated bits was proposed by Vaudenay [Vau05]. His protocol is depicted on Fig. 2.3. The sender of the message  $m$  Alice, resp. the recipient Bob, picks a random value  $R_A$ , resp.  $R_B$ . Alice sends the message  $m$  to Bob and commits on his value of  $R_A$ . Note that this protocol uses a *tag-based commitment scheme* in which the tag  $m$  is revealed, but the value  $R_A$  is kept hidden to Bob. As before, the message can be exchanged for example by email from Alice to Bob. Due to the commitment, Alice cannot change his value of  $R_A$ . Bob sends his random value  $R_B$  and finally Alice sends the decommit value.



**Figure 2.2.** The MANA I Protocol.

Both can compute its local SAS value. Alice authenticates, e.g. by telephone, the SAS value to Bob which can check whether it is valid. It is shown in [Vau05] that a 15-bit SAS using weak authentication provides pretty good security.



**Figure 2.3.** Interactive SAS-based Message Authentication.

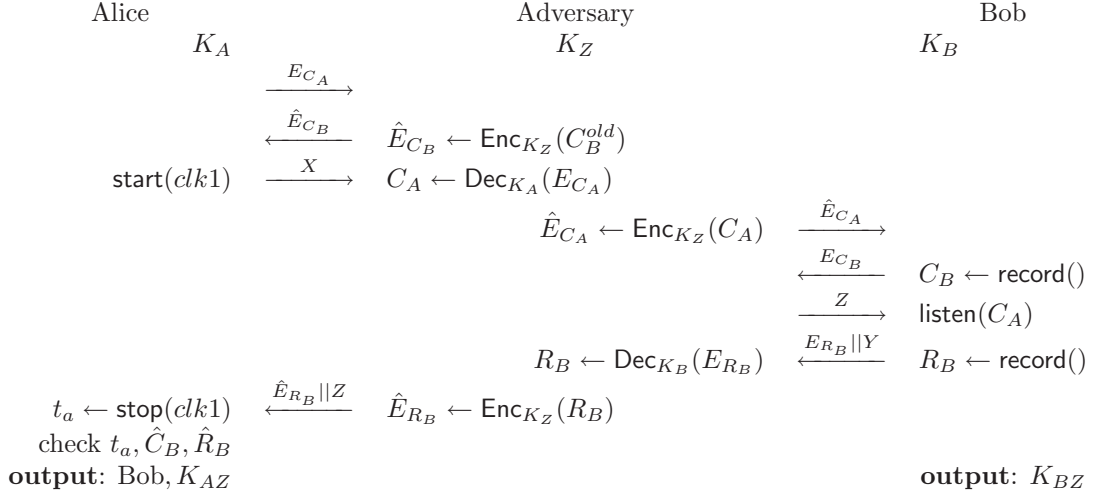
## 2.2 Protocols Using Biometric Signals

A method to setup secure communications based on vocal biometric signals was proposed by Wu-Bao-Deng [WBD05]. The protocol is depicted on Fig. 2.4. It allows Alice to authenticate Bob and to setup a secret session key. Note that it uses Diffie-Hellman values to establish the secret session key. These values are authenticated using voice signals and agreed with the help of a timer which allows to detect man-in-the-middle attacks.

In the proposed protocol, Alice, resp. Bob, chooses a value  $x$ , resp.  $y$ , and computes her Diffie-Hellman value  $X \leftarrow g^x$ , resp.  $Y \leftarrow g^y$ . Alice, resp. Bob, computes their local key by hashing their Diffie-Hellman value, i.e.  $K_A \leftarrow h(X)$ , resp.  $K_B \leftarrow h(Y)$ . Then, Alice records a “challenge”  $C_A$  which is authenticated by her voice and sends the encrypted value  $E_{C_A}$  to Bob. The encryption is done using her local key  $K_A$ . Note that this notion of “challenge” is quite trivial here. The only challenge is indeed, for Bob, to make an answer which is consistent with the “challenge” from Alice. For instance, the “challenge” can be the sentence “what did you wrote in your last mail?”. Bob makes the same operations and sends  $E_{C_B}$  to Alice. Note that each recorded sound must be as least of duration  $T$ . We stress that at this time nobody can listen to the challenges since they are encrypted and the keys are kept secret. Alice starts a clock and reveals her Diffie-Hellman value  $X$ . Thus, Bob computes  $K_A$ , i.e.  $K_A \leftarrow h(\hat{X})$ , decrypts the challenge of Alice, listens to it and checks the voice identity. If



the response  $R_B$ . Finally, the adversary answers to Alice with the response  $R_B$  (encrypted) from Bob.



**Figure 2.5.** Man-In-The-Middle Attack (Simplified Version).

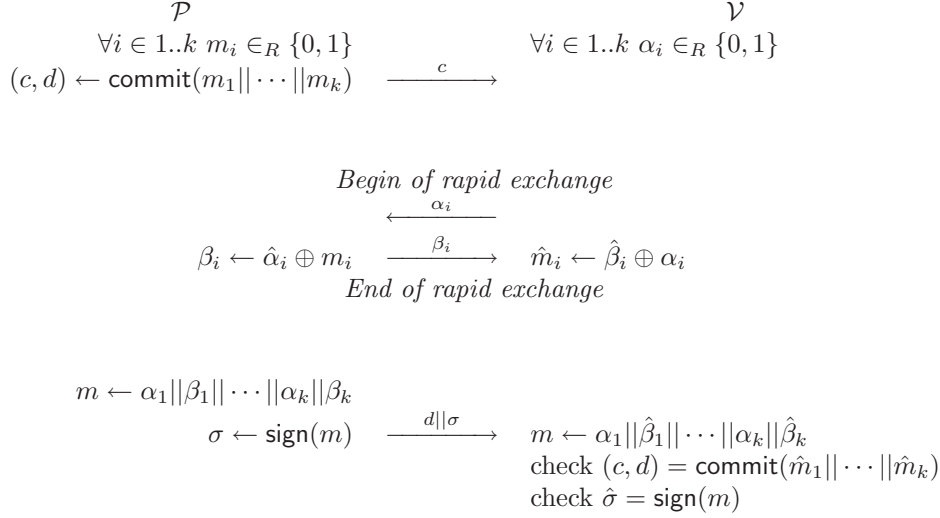
Note that the time elapsed  $t_a$  corresponds to the time for Bob to record the challenge  $C_B$ , to listen to the challenge from Alice  $C_A$ , to record the response  $R_B$ , and some duration  $\delta$  for computations and transmissions, i.e.  $t_a = |C_B| + |C_A| + |R_B| + \delta \geq 3T + \delta$ . Without attack, Bob does not have to record his challenge during this time since it has been recorded before, i.e.  $t_a = |C_A| + |R_B| + \delta \geq 2T + \delta$ . Using the timer, Alice can detect man-in-the-middle attacks. This attack shows the important role of the timer in this protocol.

We note that this protocol is very sensitive to the time in which Bob responds. Suppose Bob has made a mistake in his record sample and records another one. He has listened to the challenge of Alice and recorded two samples. Thus, the time elapsed seems to be a man-in-the-middle attack. In conclusion, this protocol is quite hard to implement in a *user-friendly* way since the timing is constraining for Bob.

### 2.3 Protocols Using Distance Bounding

Brands and Chaum [BC93] have proposed a practical method to upper-bound the physical distance between two devices. For instance, during an authentication phase between an employee and an access control, the system would like to be ensured that the employee is near, i.e. a few meters. The proposed principle is quite simple. It consists of a challenge-response using only one bit on each message. The verifier  $\mathcal{V}$  sends a bit (challenge) and the prover  $\mathcal{P}$  replies immediately with a bit (response). They assume that electronic devices which play the role of prover can have very short timings between the reception of a challenge and the sending of the corresponding response. The verifier  $\mathcal{V}$  has simply to measure the time elapsed between the sending of the challenge and the reception of the response. Knowing the time elapsed, it can easily deduce the maximal distance between them. Two attacks are described in [BC93]: the mafia fraud which is a man-in-the-middle attack where the adversary is a

fraudulent verifier at the same time as a fraudulent prover and an attack in which the prover  $\mathcal{P}$  sends bits out too soon. Solutions for preventing both attacks are proposed in [BC93] and the final protocol proposed is depicted on Fig. 2.6.



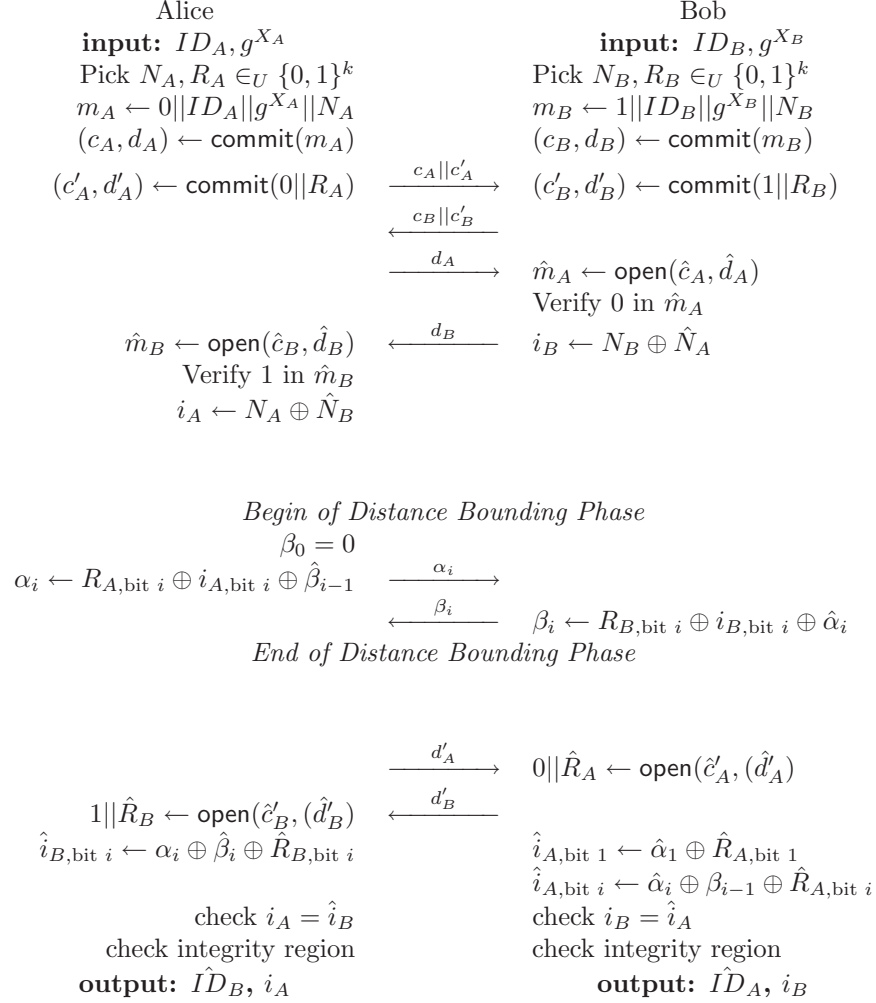
**Figure 2.6.** Distance-Bounding Protocol.

Based on this distance upper-bounding, Cagalj, Capkun and Hubaux [CCH05] have proposed a key agreement protocol for wireless network, in particular for peer-to-peer communication. In the previous protocol, it is not clear how the prover should sign the message. Cagalj, Capkun and Hubaux [CCH05] propose a method which only need authentication. The protocol is depicted on Fig. 2.7.

Brands and Chaum [BC93] proposed a method that prevents frauds where an adversary runs a man-in-the-middle attack between a legitimate prover and legitimate verifier. Frauds where a malicious prover and an adversary collaborate to cheat a verifier have been left opened. Bussard and Bagga [BB05] propose a solution for preventing both types of attacks.

Distance-bounding authentication cannot be used through the Internet since users are in general far and some others users are closer to the distant one (with very high probability). In general, this not very useful for wired links.

In conclusion, this method is not useful for authentication through a large network, but it is well adapted for local wireless networks such as mobile phones, headset or PDA's which are very close. We don't go more in detail into distance bounding protocols since we concentrate on general methods to do worldwide authentication.



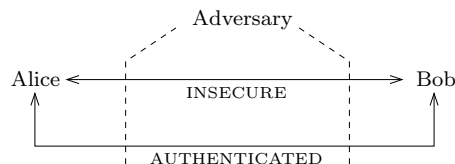
**Figure 2.7.** Key Agreement Protocol Using Distance Bounding.



# Preliminaries

This chapter explains and defines the necessary prerequisites. Definitions and formalism are used in the following chapters and thus need to be recalled. Note that the formalism used in this thesis was published by Vaudenay [Vau05].

The considered model is a communication network made up of devices which use insecure broadband communication channels between them. Note that “device” is a generic term for communication entities. For instance, a device can be a personal computer, a mobile phone, or a satellite antenna. In our model, devices can in addition use narrowband channels which can be used to authenticate short messages, i.e. short authenticated strings (SAS). We consider that a verifier Bob would authenticate a message  $m$  from a claimant Alice using the model depicted in Fig. 3.1. A device is located on a network node  $n$  of a given identity  $ID_n$ .



**Figure 3.1.** Message Authentication Protocol Channels (General) Model

In addition, it can run several instances which are formally denoted by a unique instance tag  $\pi_n^i$ .

## 3.1 Adversarial Model

We assume that adversaries have full control on the broadband communication channel. Indeed, an adversary can read messages from the channel, he can prevent a message from being delivered, he can delay it, replay it, modify it, and change its recipient address.

The security model is based on Bellare-Rogaway [BR93] which places the adversary at

the center of the network. The adversary can make queries to any instances on any nodes. He has full control on which node launches a new instance of a protocol, on the input of the protocol, and on which protocol instance runs a new step of the protocol. Namely, we assume that the adversary has access to a  $\text{launch}(n, r, x)$  oracle in which  $n$  is a node of the network,  $r$  is a character (i.e. a role to play in the protocol), and  $x$  is the input of the protocol for this character. This oracle returns a unique instance tag  $\pi_n^i$ . Since a node can a priori run concurrent protocols, there may be several instances related to the same node  $n$ . We restrict ourselves to 2-party protocols so that there are only two characters Alice and Bob in protocols. Any node can play any of these characters. The adversary also has access to oracles

- $\text{execute}(\pi_p^i, \pi_q^j)$  which runs the full protocol with a given instance pairs and returns the full transcript of protocol messages (this models passive attacks);
- $\text{send}(\pi_p^i, m)$  which sends a message  $m$  to a given instance and returns a message  $m'$  which is meant to be sent to the other participant (this models active attacks).

For example, a protocol with input  $x$  and  $y$  can be run on node  $A$  and  $B$  respectively as follows.

1.  $\pi_A^1 \leftarrow \text{launch}(A, \text{Alice}, x)$
2.  $\pi_B^1 \leftarrow \text{launch}(B, \text{Bob}, y)$
3.  $m_1 \leftarrow \text{send}(\pi_A^1, \emptyset)$
4.  $m_2 \leftarrow \text{send}(\pi_B^1, m_1)$
5.  $m_3 \leftarrow \text{send}(\pi_A^1, m_2)$
6. ...

until a message is a termination message. Note that the Bellare-Rogaway [BR93] model considers additional oracles which are shortly described below.

- $\text{reveal}(\pi_n^i)$  which reveals the session key  $sk$  to the adversary if the instance  $\pi_n^i$  have accepted them before. This query models the loss of the session key and can be used to show the consequences on others instances.
- $\text{corrupt}(n)$  which corrupts the collection of instances related to the node  $n$ . This query models the corruption of a node (all instances), for example a user-password has been stolen or a “Trojan horse” has been installed on the device on node  $n$ .
- $\text{test}$  which is specific to the semantic security of key agreement protocols. This query is used to measure the probability of success of an adversary.

These requests are not relevant in this paper since we never use long-term secrets and the output of the protocols are not secret.

By convention, we describe protocols by putting a *hat* on the notation for received messages which are not authenticated since they can differ from sent messages in the case of an active attack.

## 3.2 Authenticated Channels

When referring to “channel”, we refer by default to an insecure broadband channel without any assumption. As mentioned before, the devices can use an authenticated channel.

An *authenticated channel* is related to a node identity  $ID$ . Formally, an authenticated channel from a node  $n$  has an identifier  $ID_n$ . It allows to the recipient of a message to know the identity of the node from which the message has been sent as is. Note that an adversary cannot modify it (i.e. integrity is implicitly protected), but she can delay it, remove it, replay it, and of course read it. Precisely, an authenticated channel does not provide confidentiality.

By convention, we note  $\text{authenticate}_{ID_n}(x)$  a message  $x$  which has been sent from node  $n$  through the authenticated channel.

The **send** oracle maintains unordered sets of authenticated messages in every channel  $ID_n$  from node  $n$ . Only **send** oracles with a  $\pi_n^i$  instance can insert a new message in this set. When a **send** oracle is queried with any instance and any message  $\text{authenticate}_{ID_n}(x)$ , it is accepted by the oracle only if  $x$  is in the set related to channel  $ID_n$ . Note that concurrent or successive instances related to the same node write in the same channel, i.e. in the same set. Thus, when an instance  $\pi_n^i$  sends a message, the recipient of this message can only authenticates the node from which it has been sent, i.e.  $n$ , but not the connection to the right instance, i.e.  $i$ .

For simplicity, we assume that the input or output to the **send** oracle are either authenticated or non-authenticated messages, but not both. Namely, protocols do not concatenate authenticated and non-authenticated messages.

### 3.2.1 Weak Authenticated Channels

By default, authenticated channels without any other assumption are called *weak*. This means that an adversary can delay a message, remove it, or replay it. In particular, the owner of the message has not the insurance that the message has been delivered to the recipient.

### 3.2.2 Stronger Authenticated Channels

In some cases we need special assumptions on the authenticated channel. Thus, we can consider *stronger authenticated channels*, namely channels in which additional properties are achieved. In the following, we propose some possible properties that can be assumed on a stronger authentication channel.

**Stall-free transmission** assumes that when a message is released by a **send** oracle either it is used as input in the just following **send** oracle query (either authenticated or not) or it is never used.

**Transmission with acknowledgment** assumes that messages are released with a destination node identifier and the sender can check whether an instance at the destination node has received the message or not.

**Listener-ready transmission** assumes that the sender can check if an instance at the destination node is currently ready to listen to the authenticated channel.

	Interactive		Non-interactive		
	Encounter	Telephone	Mail	Email	Voice record
Stall-free	✓	✓			
Acknowledgment	✓	✓			
Listener-ready	✓	✓			
Immediate delivery	✓				

**Figure 3.2.** Stronger Properties of Human Authenticated Channels

**Transmission with immediate delivery** assumes that an input message of a `send` oracle is immediately delivered to the recipient.

### 3.2.3 Examples

As seen in Section 1.1, a **face to face conversation** and a **telephone call** achieve authenticity. In addition, these channels achieve some of the stronger properties described above. Suppose two persons would communicate. When the first person starts talking, he knows that the second one is listening (listener-ready). When one talks to the other one, he know that the message has not been recorded since interactivity implies coherent conversations (stall-free). Humans can also sense if the other one has listened to the message (acknowledgment). Finally, when a person talks face to face with another one, he has the insurance that the another one has received immediately the sound (immediate delivery). Using telephone, this is not the case: sometimes we feel a delay between the moment when one starts to speak and the other hears. Often this delay brings collisions of voices.

A **mail** can be stalled and released in a different order. The sender has no confirmation in general that the mail has been received (except using a registered mail). Finally, the recipient may not be ready to receive it. Thus, a conventional mail achieves none of these properties and a registered mail is a transmission with acknowledgment only.

An **electronic mail** is worst in term of security as a mail and also achieves none of these properties. Note that an email without any cryptographic appendix tools such as a GPG signature is in fact not an authenticated channel since it can easily be forged.

A **voice record** achieves none of these properties since the message could be a recorded one. The recorder has no confirmation that the destination has heard it. The recipient is in general not ready to listen.

It is clear that mail, electronic mail and voice record are not delivered immediately.

## 3.3 Message Authentication Protocols

A *message authentication protocol* has an input  $m$  on the side of the claimant Alice and nothing on the side of the verifier Bob. It has an output  $\hat{ID} \parallel \hat{m}$  on Bob's side and nothing on Alice's side. Authentication is successful if the output is  $\hat{ID} = ID_A$  and  $\hat{m} = m$ , meaning that  $\hat{m}$  coming to Bob was authenticated as sent by a node of identity  $ID_A$ .

We call a protocol *non-interactive* if only it uses messages send by Alice to Bob. Otherwise, we say that the protocol is *interactive*.

On a global perspective, several  $\text{launch}(A_k, \text{Alice}, m_k)$  and several  $\text{launch}(B_\ell, \text{Bob}, \emptyset)$  can be queried. These queries create several  $\pi_{A_k}^{i_k}$  instances of Alice (authentication claims) and several  $\pi_{B_\ell}^{j_\ell}$  instances of Bob (authentication verifications). We may have a perfect matching between the  $k$ 's and  $\ell$ 's such that related instances have matching conversations which fully follow the protocol specifications, and the  $\pi_{B_\ell}^{j_\ell}$  ends with output  $\text{ID}_{A_k} || m_k$  for the matching  $k$ . In any other case, we say that an attack occurred.

**Attack Definition.** We say that an attack is successful if there exists at least an instance  $\pi_{B_\ell}^{j_\ell}$  which terminated and output  $\hat{ID} || \hat{m}$  such that there is no  $k$  for which  $\hat{ID} = \text{ID}_{A_k}$  and  $\hat{m} = m_k$ . Note that many protocol instances can endlessly stay in an unterminated state or turn in an abort state.

We call *one-shot attacks* the attacks which launch a single instance of Alice and Bob. We call *attacks* the general attacks that can use several instances of Alice and Bob. The *attack cost* is measured by

- the number  $Q$  of launched instances of Alice and Bob, i.e. the *online complexity*.
- the additional complexity  $C$ , i.e. the *offline complexity*.
- the probability of success  $p$ .

An attack of cost  $(Q, C, p)$  is cheaper than an attack of cost  $(Q', C', p')$  if  $Q \leq Q'$ ,  $C \leq C'$ , and  $p \geq p'$ , but note that some attacks are incomparable in terms of cost without any more implementation assumptions.

Here is a useful lemma taken from [Vau05].

**Lemma 1.** *We consider a message authentication protocol with claimant Alice and verifier Bob in which a single SAS is sent. We denote by  $\mu_A$  (resp.  $\mu_B$ ) the complexity of Alice's (resp. Bob's) part. We consider adversaries such that the number of instances of Alice (resp. Bob) is at most  $Q_A$  (resp.  $Q_B$ ). We further denote  $T_0$  and  $p_0$  their time complexity and probability of success, respectively. There is generic transformation which, for any  $Q_A$ ,  $Q_B$ , and any adversary, transforms it into a one-shot adversary with complexity  $T \leq T_0 + \mu_A Q_A + \mu_B Q_B$  and probability of success  $p \geq p_0 / Q_A Q_B$ .*

Assuming that no adversary running a one-shot attack has a probability of success larger than  $p$ , using Lemma 1, we can upper bound the probability of success of an attack which uses  $Q_A$ , resp.  $Q_B$ , instances of Alice, resp. Bob. More precisely, it has a probability of success at most  $Q_A Q_B p$ .

### 3.4 Commitment Schemes

A commitment scheme can be seen as a “locked combination safe”. When Alice would commit on a message  $m$ , she placed  $m$  into the “safe” and closed it. The safe is also the commit object  $c$  and can be given to another party. The message is revealed only when the combination is known, i.e. the decommit  $d$  is revealed. Obviously, the message  $m$  cannot be known by

others party prior its opening, i.e. the “locked safe” is “hiding”, and cannot be modified by Alice, i.e. the “locked safe” is “binding”.

We can formalize a commitment scheme by two algorithms `commit` and `open`. For any message  $m$  we have  $(c, d) \leftarrow \text{commit}(m)$ . The  $c$  value is called the *commit* value and the  $d$  value the *decommit* value. Knowing both  $c$  and  $d$ , the message can be recovered using the `open` oracle, i.e.  $m \leftarrow \text{open}(c, d)$ . As a “locked safe”, a commitment scheme should be *hiding*, meaning that for any  $c$ , it is hard to deduce any information about the corresponding message  $m$ , and *binding*, meaning that one cannot find  $c, d, d'$  such that  $(c, d)$  and  $(c, d')$  open to two different messages.

We also introduce *keyed commitment schemes* which have in addition a `setup` oracle to initialize a pair of keys, i.e.  $(K_p, K_s) \leftarrow \text{setup}()$ . The public key  $K_p$  is used in `commit` and `open` oracles. Note that  $K_p$  may be empty.

### 3.4.1 Tag-Based Commitment Model

Tag-based commitment schemes are particular commitments schemes in which the message  $x$  is composed of two parts: a known part  $m$  called tag and an hidden part  $r$ . Let  $k$  be the bit length of the hidden value  $r$ . The `setup` algorithm is the same as for tag-less commitments, but the two algorithms `commit` and `open` are redefined. The three algorithms are finally defined as follows.

**Setup algorithm.** It yields a pair of keys  $(K_p, K_s)$ .

**Commit algorithm.** For any pair of keys  $(K_p, K_s)$ , any tag  $m$ , and any value  $r$  we have  $(c, d) \leftarrow \text{commit}(K_p, m, r)$ .

**Open algorithm.** From  $K_p$ ,  $c$ ,  $d$  and the tag  $m$ , the hidden value  $r$  can be recovered using the `open` oracle, i.e.  $r \leftarrow \text{open}(K_p, m, c, d)$ .

Standard commitment schemes, as previously defined in which the message  $m'$  is kept hidden, can be constructed using a tag-based commitment scheme with an empty tag, i.e.  $m = \perp$ , and an hidden value equals to the conventional message, i.e.  $r = m'$ . We have finally  $x = m || r = m'$ .

### 3.4.2 Completeness, Hiding, and Binding Properties

We define here the completeness, hiding and binding properties for tag-based commitment schemes.

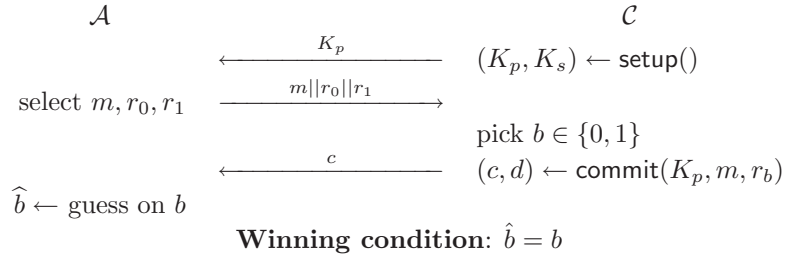
Any commitment scheme must clearly satisfy the *completeness property*, i.e. a commit value  $c$  and a decommit value  $d$  which has been yield by the `commit` algorithm with input  $m$  and  $r$  would open to  $r$ . More formally, the completeness property is defined as follows.

**Completeness property.** For any pair of keys  $(K_p, K_s)$ , any tag  $m$ , any value  $r$ , and any  $(c, d) \leftarrow \text{commit}(K_p, m, r)$ , we have  $r = \text{open}(K_p, m, c, d)$

In addition, commitments schemes should be *hiding*, i.e. the commit value  $c$  reveals no information about the hidden value  $r$ . More formally, the hiding property is defined as follows.

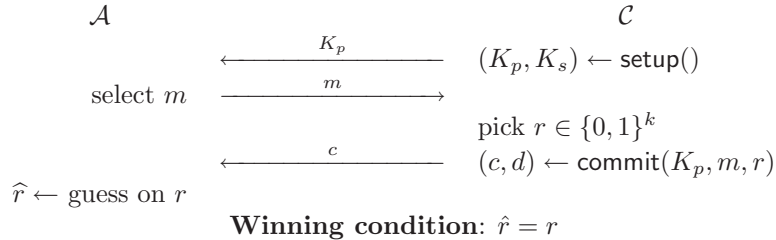
**Hiding property.** For any pair of keys  $(K_p, K_s)$ , any tag  $m$ , any values  $r_0, r_1$ , and any  $(c, d) \leftarrow \text{commit}(K_p, m, r_b)$ ,  $c$  gives no information on the random binary value  $b$ .

We defeat the hiding property with the semantic hiding (SH) game which is described on Fig. 3.3: the adversary  $\mathcal{A}$  selects a tag  $m$  and two hidden values  $r_0$  and  $r_1$  and sends them to the challenger  $\mathcal{C}$ . The challenger flips an unbiased coin  $b$  and commits to  $(m, r_b)$ . Given the commit value the adversary guesses  $b$  and succeeds if the guess is correct. The scheme is  $(T, \epsilon_h)$ -semantically hiding if any adversary  $\mathcal{A}$  bounded by a complexity  $T$  has a probability of success at most  $\frac{1}{2} + \epsilon_h$ .



**Figure 3.3.** SH Game.

We also can defeat the hiding property with the full hiding (FH) game which is described on Fig. 3.4: The adversary picks a tag  $m$  and sends it to the challenger  $\mathcal{C}$ .  $\mathcal{C}$  picks a hidden value  $r \in \{0, 1\}^k$  and commits on  $(m, r)$ .  $\mathcal{C}$  reveals the value  $c$  to the adversary  $\mathcal{A}$ . The adversary guesses  $r$  and succeeds if the guess is correct. The scheme is  $(T, \epsilon_h)$ -fully hiding if any adversaries  $\mathcal{A}$  bounded by a complexity  $T$  has a probability of success at most  $2^{-k} + \epsilon_h$ .



**Figure 3.4.** FH Game.

Note that a scheme is perfectly hiding if it is  $(\infty, 0)$ -hiding.

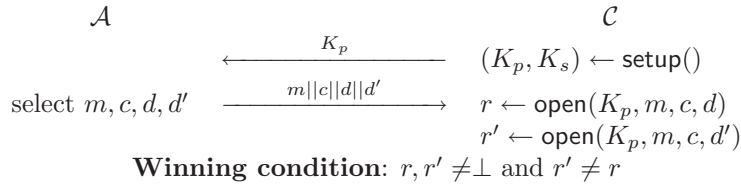
**Lemma 2** ([Vau05]). *There exists a (small) constant  $v$  such that for any  $T$  and  $\epsilon_h$ , a  $(T + v, \epsilon_h)$ -semantically hiding scheme is a  $(T, 2\epsilon_h)$ -fully hiding commitment scheme.*

Obviously, a  $(T + v, \epsilon_h)$ -fully hiding commitment scheme is  $(T, \epsilon_h)$ -semantically hiding. Hence, the two notions of hiding commitment schemes are essentially equivalent.

In addition, commitments should be *binding*, i.e. an adversary which has committed on a value with  $c$  can not opens to two different hidden values  $r_0$  and  $r_1$ . More formally, the binding property is defined as follows.

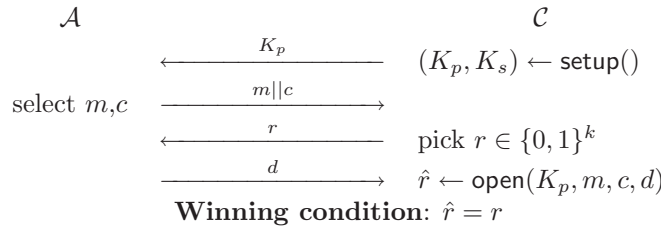
**Binding property.** For any pair of keys  $(K_p, K_s)$  and any adversary  $\mathcal{A}$  bounded by a complexity  $T$ , there exists a small  $\epsilon$  such that  $\mathcal{A}$  cannot find  $(m, r_0, r_1, c, d_0, d_1)$  where  $\text{open}(K_p, m, c, d_b)$  yields  $r_b$  respectively for  $b = 0$  and  $b = 1$  and  $r_0 \neq r_1$  with probability bigger than  $\epsilon$ .

The semantic binding (SB) game of Fig. 3.5 must be hard, i.e. for any tag  $m$  and any commit value  $c$ , it is hard to find two decommit values  $d$  and  $d'$  such that  $r \leftarrow \text{open}(K_p, m, c, d)$  and  $r' \leftarrow \text{open}(K_p, m, c, d')$  with  $r \neq r'$ . The scheme is  $(T, \epsilon_b)$ -semantically binding if given  $m$  and  $c$  any adversaries  $\mathcal{A}$  bounded by a complexity  $T$  has a probability to find two decommit values  $d$  and  $d'$  which is at most  $\epsilon_b$ .



**Figure 3.5.** SB Game.

We can also define the full binding (FB) game. As described on Fig. 3.6, it works as follows: the adversary  $\mathcal{A}$  selects a tag  $m$  and a commit value  $c$ . Then, he sends it to the challenger  $\mathcal{C}$ .  $\mathcal{C}$  picks a random value  $r$  and sends it to  $\mathcal{A}$ . The adversary  $\mathcal{A}$  proposes a decommit value  $d$  and succeeds if it opens to  $r$ , i.e.  $r = \text{open}(K_p, m, c, d)$ . The scheme is  $(T, \epsilon_b)$ -fully binding if any adversaries  $\mathcal{A}$  bounded by a complexity  $T$  has a probability of success at most  $2^{-k} + \epsilon_b$ .



**Figure 3.6.** FB Game.

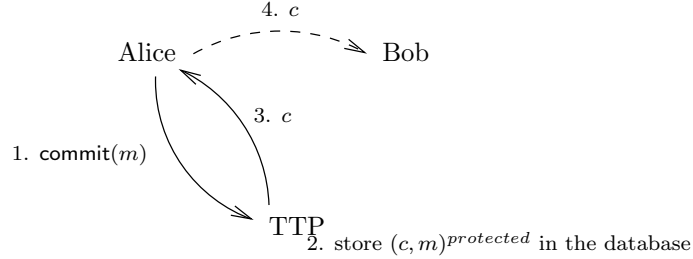
Note that a scheme is perfectly binding if it is  $(\infty, 0)$ -binding.

### 3.4.3 Ideal Commitment Model

The notion of *ideal commitment model* describes a scheme which is perfectly hiding and perfectly binding.

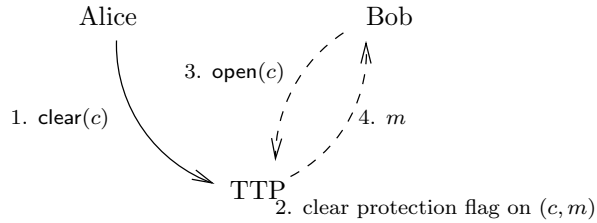
For instance, an ideal commitment scheme can be implemented using a trusted third party (TTP) as follows.

The  $\text{commit}(m)$  algorithm consists of sending the message  $m$  securely to the TTP. The TTP binds it to a unique commit value  $c$ , inserts  $(c, m)$  in a database with a protection flag and returns  $c$  to the owner. There is no decommit value.



**Figure 3.7.** Ideal Commitment: Commit Algorithm.

The  $\text{open}(c)$  algorithm is in fact a call to the TTP. The TTP clears the protection flag of  $(c, m)$  which becomes available for anyone.



**Figure 3.8.** Ideal Commitment, Decommit Algorithm.

Ideal commitment can be also implemented using the notion of universal composable commitment schemes as proposed by Damgård and Nielsen in [DN02]. Canetti and Fischlin in [CF01] propose another commitment scheme which behaves like an “ideal commitment service”. It is based on the *common reference string model* (CRS), see after, where all parties uses or can access to the same common string.

### 3.4.4 Extractable and Equivocable Commitment Schemes

We define an *extractable commitment scheme* as an extension of a general tag-based commitment scheme in which there is an additional deterministic algorithm  $\text{extract}$ . With inputs  $m$  and  $c$ , the  $\text{extract}$  algorithm yields a  $r$  value, i.e.  $r \leftarrow \text{extract}(K_s, m, c)$  whenever there exists  $d$  which opens to  $r$  using  $r \leftarrow \text{open}(K_p, m, c, d)$ . Note that extractable commitments are perfectly binding since for a given  $m$  and a given  $c$ , only one  $r$  can be yield by the  $\text{extract}$  algorithm. Thus, there exists only one  $d$  and any adversary loses the SB game with probability 1.

We define an *equivocable commitment scheme* as an extension of a tag-based commitment scheme in which there are two additional algorithms  $\overline{\text{commit}}$  and  $\text{equivocate}$ . With input  $m$ , the  $\overline{\text{commit}}$  algorithm yields a fake commit value  $c$  and an information value  $\xi$  using the secrete key  $K_s$ , i.e.  $(c, \xi) \leftarrow \overline{\text{commit}}(K_s, m)$ . Using  $K_s$ , and with inputs  $(m, r, c, \xi)$  where  $c$  and  $\xi$  are outputs of  $\overline{\text{commit}}$  algorithm, the  $\text{equivocate}$  algorithm yields a decommit value

$d \leftarrow \text{equivocate}(K_s, m, r, c, \xi)$  such that  $(m, c, d)$  opens to the  $r$  value. Assume that for any  $K_p || K_s$  and any  $m$ , the distribution of fake commit values are identical to the distribution of real commit values for any  $r$ , equivocable commitments are perfectly hiding since a  $c$  can be generated before and independently of the hidden value  $r$ .

Adversaries playing the SH, FH, SB, or FB games may have access or not to oracles. For extractable commitments, they may query an  $\text{extract}(K_s, \cdot, \cdot)$  oracle, except on the target tag  $m$ . (Note that they have no access to  $K_s$ , but the oracle has.) For equivocable commitments, they may query  $\overline{\text{commit}}(K_s, \cdot)$  and the related  $\text{equivocate}(K_s, m, \cdot, c, \xi)$  oracles, except on the target tag  $m$ , but cannot see the value  $\xi$ . (Namely: an oracle  $\text{equivocate}(K_s, m, \cdot, c, \xi)$  is created by  $\overline{\text{commit}}(K_s, \cdot)$  so that  $K_s$  and  $\xi$  remain hidden to the adversary.)

### 3.4.5 Trapdoor Commitment Model

The notion of *trapdoor commitment* was introduced by Brassard-Chaum-Crepeau [BCC88]. We define (tag-less)  $(T, \epsilon)$ -*trapdoor commitment schemes* by four algorithms  $\text{setup}$ ,  $\text{commit}$ ,  $\text{open}$ , and  $\text{equivocate}$ . The first three work as before. The algorithm  $\text{equivocate}$  defeats the binding property by using the secret key  $K_s$ . There is no fake commitment algorithm. Indeed, the algorithm  $\text{equivocate}$  needs no additional information except  $K_s$ . Note that this primitive is a particular case of *strongly equivocable commitment* as described by Damgård-Groth [DG03].

More precisely, for any  $(K_p, K_s) \leftarrow \text{setup}()$ , a trapdoor commitment scheme is such that

**Commitment properties.** The algorithms  $\text{setup}$ ,  $\text{commit}$  and  $\text{open}$  form a tag-less commitment scheme which satisfies the completeness property, which is perfectly hiding, and which is  $(T, \epsilon)$ -binding.

**Trapdoor property.** For any pair of keys  $(K_p, K_s)$  and any message  $m$ , the two distributions

$$(c, d) \leftarrow \text{commit}(K_p, m)$$

and

$$(c \in_U \mathcal{C}, d \leftarrow \text{equivocate}(K_s, m, c))$$

are indistinguishable.

For instance, a trapdoor commitment based on the discrete logarithm problem was proposed by Boyar and Kurtz [BK90]. Another trapdoor commitment scheme was proposed by Catalano et al. [CGHGN01] which is detailed in the next section.

Trapdoor commitment schemes are perfectly hiding and computationally binding commitment schemes. Note that it is impossible to distinguish whether a committed value and its corresponding decommit value, i.e.  $(c, d)$  have been yield by using the standard  $\text{commit}$  algorithm or by using the trapdoor, the  $\text{equivocate}$  algorithm, and choosing  $c$  uniformly at random.

In the Damgård-Groth construction, adversaries can query  $\text{equivocate}(K_s, \cdot, \cdot, \cdot)$  oracle (except on the selected  $c$ ). In the Boyar-Kurtz and Catalano et al. constructions, they cannot.

### 3.4.6 Examples

**Extractable Commitment Model Using a Random Oracle.** Extractable commitment schemes can be designed from a random oracle  $H$ . Such a schemes were defined in Pass [Pas03]. The random oracle  $H$  yields from any input  $d$  a value  $c \leftarrow H(d)$ , with  $c \in \{0, 1\}^h$ . The  $\text{setup}()$  algorithm actually returns no key at all.

The  $\text{commit}(m, r)$  algorithm with input hidden value  $r$  in  $\{0, 1\}^k$  picks a random value  $e$  in  $\{0, 1\}^\ell$ , builds  $d \leftarrow r||e$ , and call the random oracle  $c \leftarrow H(m||d)$ .

The  $\text{open}(m, c, d)$  algorithm simply extracts  $r$  from  $d$  and checks that  $c = H(m||d)$ .

An  $\text{extract}(m, c)$  oracle can be added to this commitment scheme. Indeed, when there was no collision on the outputs of the oracle  $H$ , the hidden value corresponding to a given tag  $m$  and commit value  $c$  queried to the oracle  $H$  can be extracted from the history of  $H$  queries.

Note that by allowing  $q$  queries to the oracle  $H$ , the scheme is  $(\infty, q \cdot 2^{-\ell-k})$ -extractable with probability at least  $1 - q^2 \cdot 2^{-h-1}$ . This scheme is pretty safe with  $h = 2\ell$  and  $k = 80$  (see e.g. [Vau05]) and the oracle  $H$  can be typically an hash function, e.g. SHA-1 [SHA93, SHA95].

**Commitment Scheme in the Common Reference String Model.** We assume all implementations to use the same trusted public key, a common reference string (CRS), and believe that no corresponding secrete key is kept by anyone. Note that the use of the common public key can be “hard-coded” or can be an access. As said before, Canetti and Fischlin [CF01] propose a commitment scheme based on the CRS model which behaves like “ideal commitment scheme”. All commitment schemes below are in the CRS model.

**Paillier-based Trapdoor Commitment Scheme.** A trapdoor commitment scheme was proposed by Catalano et al. [CGHGN01] based on the Paillier’s trapdoor permutation [Pai99]. The proposed scheme is tag-less. It uses a RSA modulus  $N = pq$  and a value  $h \in \mathbb{Z}_{N^2}$  such that its order is a multiple of  $N$ . The public key is  $K_p \leftarrow (N, h)$  and the private key is  $K_s \leftarrow (p, q, h)$ .

The  $\text{setup}()$  algorithm outputs the two keys,  $K_p$  and  $K_s$ , as described previously.

The  $\text{commit}(K_p, m)$  algorithm of a message  $m \in \mathbb{Z}_N$  picks uniformly two random values  $r, s$  and outputs  $c \leftarrow (1 + mN)r^N h^s \bmod N^2$  and  $d \leftarrow (r, s)$ . Note that the commit value  $c$  is uniformly distributed for any  $m$  since  $r$  and  $s$  are uniformly distributed and  $(r, s) \mapsto r^N h^s \bmod N^2$  is the Paillier trapdoor permutation (see [Pai99]). We denote  $\mathcal{F}_h(r, s)$  this permutation.

The  $\text{open}(K_p, c, d)$  algorithm yields  $m$  by solving  $c = (1 + mN)r^N h^s \bmod N^2$  from  $d = (r, s)$  and  $K_p = (N, h)$ .

The  $\text{equivocate}(K_s, m, \hat{c})$  algorithm uses the collision-finding function, i.e. given a commit  $\hat{c}$  and a message  $m$ , one can find  $\hat{d} \leftarrow (\hat{r}, \hat{s})$  such that  $\hat{c} = (1 + mN)\mathcal{F}_h(\hat{r}, \hat{s}) \bmod N^2$  by using the trapdoor on the Paillier permutation and knowing  $p, q$ , i.e.  $(\hat{r}, \hat{s}) \leftarrow \mathcal{F}_h^{-1}(\hat{c}(1 + mN)^{-1})$ . Thus, given a  $\hat{c}$ , an adversary can find  $\hat{d}$  for any message  $m$  and thus defeats the binding property.

**Pedersen Commitment Scheme.** An interesting and simple tag-less commitment scheme was proposed by Pedersen in [Ped91].

The  $\text{setup}()$  phase consists of choosing two prime numbers  $p$  and  $q$  such that  $q$  divides  $p-1$ . Let  $G_q$  the unique subgroup of order  $q$  of  $\mathbb{Z}_p^*$  and let  $g$  a generator of  $G_q$ . In addition, it picks a random  $y \in G_q$  such that nobody knows  $\log_g y$ . The public parameters are  $(p, q, g, y)$ . Note that the setup phase can be done by a trust third party.

The  $\text{commit}(K_p, m)$  algorithm with input message  $m \in \mathbb{Z}_q$  starts by picking a random  $\ell \in G_q$  and then computes  $c \leftarrow g^m y^\ell$ . The decommit value is simply  $d \leftarrow (m, \ell)$ .

The  $\text{open}(K_p, c, d)$  algorithm has just to check that  $c = g^m y^\ell$ .

This scheme is perfectly hiding since  $y^\ell$  is a random value and thus  $c$  reveals no information about  $m$ . On the other hand, this scheme is computationally binding. Indeed, an adversary who can win against the tag-less SB game proceeds as follows: He discloses some  $c, d, d'$  such that  $d = (m, \ell)$ ,  $d' = (m', \ell')$ ,  $m \neq m'$ , and  $c = g^m y^\ell = g^{m'} y^{\ell'}$ . The secret parameter leaks by  $\log_g(h) = \frac{m-m'}{\ell'-\ell}$ .

**A Tag-based Equivocable Commitment Scheme.** We present here a tag-based trap-door commitment scheme from MacKenzie and Yang [MY04]. The scheme is based on the Pedersen commitment scheme [Ped91] and on the DSA signature scheme [DSS94, DSS00].

The  $\text{setup}()$  algorithm generates a DSA pair of keys  $(K_p, K_s)$ , i.e.

$$K_p = (g, p, q, y), \quad K_s = x.$$

The  $\text{commit}(K_p, m, r)$  algorithm with input  $r$  in  $\mathbb{Z}_q$  picks a  $k$  and computes  $g'$  and  $h$  as follows.

$$k \in_U \mathbb{Z}_q, \quad g' \leftarrow g^k \bmod p, \quad h \leftarrow g^{H(m)} y^{g'} \bmod p.$$

Note that  $(r_{DSA}, s_{DSA})$  is a DSA signature of  $m$  where  $r_{DSA} = g' \bmod q$  and  $s_{DSA} = \log_{g'}(h)$ , i.e.  $s_{DSA} = \frac{H(m) + xr}{k}$ .

Then, it uses the Pedersen commitment scheme to commit on  $r$  using  $(g', h)$  as public parameters, i.e. it picks  $\ell$  and computes  $c'$  as follows.

$$\ell \in \mathbb{Z}_q, \quad c' \leftarrow (g')^\ell h^r.$$

Finally, the commit value  $c$  and the decommit value  $d$  are the following.

$$c \leftarrow (g', c'), \quad d \leftarrow (r, \ell)$$

The  $\text{open}(K_p, m, c, d)$  algorithm computes  $h$  as follows.

$$h \leftarrow g^{H(m)} y^{g'} \bmod p$$

Then, it checks that

$$c' = (g')^\ell h^r$$

and if the condition is verified it outputs  $r$ , otherwise it outputs  $\perp$ .

These three first algorithms form a commitment scheme. In addition, this scheme as a  $\text{commit}(K_s, m)$  algorithm which use the secrete key  $K_s$  to yield a fake commit value and additional information which are required by the equivocate algorithm.

The  $\overline{\text{commit}}(K_s, m)$  algorithm computes a DSA signature  $\sigma_{DSA} \leftarrow (r_{DSA}, s_{DSA})$  on  $m$  using the secret key  $K_s$ , i.e.

$$k \in \mathbb{Z}_q^*, \quad r_{DSA} \leftarrow g^k \bmod p \bmod q, \quad s_{DSA} \leftarrow \frac{H(m) + xr_{DSA}}{k} \bmod q.$$

Then, it computes  $g'$  and  $h$  as

$$h \leftarrow g^{H(m)} y^{r_{DSA}} \bmod p, \quad g' \leftarrow h^{s_{DSA}^{-1}} \bmod p$$

and picks  $\ell$  and computes  $c'$  as

$$\ell \in \mathbb{Z}_q, \quad c' \leftarrow h^\ell \bmod p.$$

Finally, the fake commit value  $\hat{c}$  and the auxiliary information  $\xi$  are

$$\hat{c} \leftarrow (g', c'), \quad \xi \leftarrow (\ell, s_{DSA}).$$

Note that  $g'$  is in fact equal to  $g^k \bmod p$ .

The  $\text{equivocate}(K_s, m, \hat{r}, \hat{c}, \xi)$  algorithm outputs a fake decommit value  $\hat{d}$  such that  $\text{open}(K_p, m, \hat{c}, \hat{d})$  yields  $\hat{r}$ . Thus, it must find a  $\hat{d}$ , i.e. a  $\hat{\ell}$  (since  $\hat{r}$  is given), such that

$$c' = (g')^{\hat{\ell}} h^{\hat{r}} \bmod p$$

which can be written as

$$c' = (g')^{\hat{\ell}} g^{\hat{r}H(m)} y^{\hat{r}g'} \bmod p.$$

But, we know that  $y = g^x \bmod p$  and  $g' = g^k \bmod p$  and we obtain

$$c' = g^{k\hat{\ell} + \hat{r}H(m) + x\hat{r}g'} \bmod p.$$

In reality, the  $c'$  value yields by the  $\overline{\text{commit}}$  algorithm is

$$c' = h^\ell \bmod p = g^{H(m)\ell + xr_{DSA}\ell} \bmod p.$$

Recall from DSA that  $g$  is of order  $q$ . Thus, we can deduce that

$$k\hat{\ell} + \hat{r}H(m) + x\hat{r}g' = H(m)\ell + xr_{DSA}\ell \bmod p - 1.$$

Note that  $r_{DSA} = g' \bmod q$ . We have

$$\hat{\ell} = (\ell - \hat{r}) \frac{H(m) + xr_{DSA}}{k} \bmod q$$

Consequently, we obtain

$$\hat{\ell} = (\ell - \hat{r}) s_{DSA} \bmod q$$

and this is the reason why  $\ell$  and  $s_{DSA}$  are in the extra information  $\xi$ .

Note that this scheme has a stronger binding property called *simulation sound binding property* which guarantees that a commitment made by an adversary with tag  $m$  is binding even if he has seen many *simulated* commit values but never a commitment using  $m$ . It is showed in [MY04] that if an adversary can break this property then it can also break DSA.

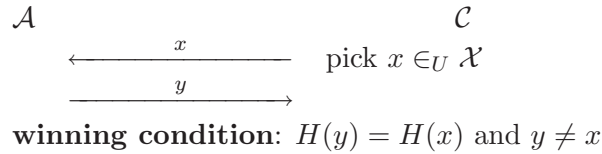
## 3.5 Hash functions

### 3.5.1 Collision-Resistant Hash Functions

A collision-resistant hash function (CRHF) is a hash function in which it should be hard to find two inputs  $x$  and  $y$  such that  $H(x) = H(y)$  and  $x \neq y$ . Due to the birthday attacks, the output space must be at least of  $2^{160}$ .

### 3.5.2 Weakly Collision-Resistant Hash Functions

Weak collision resistance means that the game of Fig. 3.9 is hard. Assume a  $(T, \epsilon)$ -weakly collision-resistant hash function (WCRHF)  $H$  defined on a finite set  $\mathcal{X}$ . For any adversary  $\mathcal{A}$  bounded by a complexity  $T$ ,  $\mathcal{A}$  wins the WCR game on Fig. 3.9 with probability smaller than  $\epsilon$ .



**Figure 3.9.** WCR game.

### 3.5.3 Universal Hash Functions Families

An  $\epsilon$ -universal hash function family (UHFF) is a collection of functions  $H_K$  from a message space to a finite set  $\{0, 1\}^k$  which depends on a random parameter  $K$  such that for any  $x \neq y$  we have

$$\Pr[H_K(x) = H_K(y)] \leq \epsilon.$$

where the probability is over the random selection of  $K$ .

# On the Required Entropy of Authenticated Communications

In this chapter, we would like to upper bound the security of an arbitrary message authentication protocol  $P$  given the amount of authenticated strings it uses. Assume that the protocol is used between Alice and Bob. We consider the more general case by supposing that the protocol can use any sequence of authenticated messages in a given set  $S$  during the protocol. We call it a *transcript*. Note that authenticated strings are interleaved with regular messages which are not represented in the transcript. For any input message  $m$ , the used transcript during a protocol instance is picked in the set  $S$  of all possible transcripts with a distribution  $\mathcal{D}_m$ .

First, we analyze the security against one-shot adversaries, which can only use one instance of Alice and one instance of Bob.

Second, we consider adversaries which can use many instances of Alice and Bob. Using a weak authenticated channel, adversaries can delay or replay authenticated messages. With protocols using a single  $k$ -bit SAS, we have the following attacks.

**Delay attack.** An attacker can start a protocol with Alice to recover an authenticated SAS. Then he can launch several protocols with Bob until a SAS matches and then deliver the SAS to this instance of Bob.

**Catalog attack.** Similarly, an attacker can launch several instances of Alice and use an authenticated SAS to pass the authentication with Bob. This attack works when the SAS catalog is close to the complete one.

**Trade-off attack.** We can further trade the number of Bob's instances against the number of Alice's instances and have a birthday paradox effect.

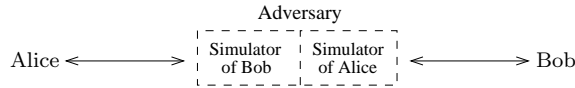
Note that the first two attacks work within a number of trials around  $2^k$  but the third one needs only around  $2^{k/2}$  trials.

## 4.1 A Generic One-Shot Attack

**Theorem 1.** *We consider an arbitrary message authentication protocol between Alice and Bob which uses an authentication channel. Let  $S$  be the set of all possible protocol transcripts through the authentication channel for any input message. Let  $s$  be its cardinality. There exists a generic one-shot attack with probability of success greater than  $\frac{1}{s} - 2^{-t}$  which runs in polynomial time in terms of  $t$  (with equality  $\frac{1}{s} - 2^{-t}$  if and only if the SAS distribution is uniform among  $S$ ).*

Theorem 1 says that there exists a one-shot attack against any message authentication protocol which succeeds with probability at least essentially  $\frac{1}{s}$  where  $s$  is the size of  $S$ . Thus, any message authentication protocol which has no better one-shot attack is essentially optimal since any other protocol can be attacked using this generic attack and consequently can not have a better security. In addition, Theorem 1 says that a protocol is more resistant to one-shot attacks when its SAS distribution is uniform.

*Proof.* We consider a general man-in-the-middle attack in which the adversary first picks  $m \in_U \{0, 1\}^t$  and  $\hat{m} \in_U \{0, 1\}^t$  and launches Alice with input  $m$ . The attack runs synchronized protocols between Alice and a simulator for Bob, and a simulator for Alice with input  $\hat{m}$  and Bob as depicted on Fig. 4.1. Following the attack, every authenticated message which



**Figure 4.1.** Generic One-Shot Attack

must be sent by the simulator is replaced by an authenticated message which has just been received by the simulator.

Let  $SAS_m$  be the (random) sequence of all authenticated strings (the transcript) which would be exchanged in the protocol between Alice and the simulator if the simulator were honest, and  $SAS_{\hat{m}}$  be the similar sequence between the simulator and Bob. Clearly, if  $SAS_{\hat{m}} = SAS_m$ , the attack succeeds. Note that an attack makes sense only if  $\hat{m}$  is different of  $m$ .

We have

$$\begin{aligned} \Pr[\text{success}] &= \Pr[SAS_m = SAS_{\hat{m}} \text{ and } m = \hat{m}] \\ &\geq \Pr[SAS_m = SAS_{\hat{m}}] - \Pr[m = \hat{m}]. \end{aligned}$$

Note that  $SAS_m$  and  $SAS_{\hat{m}}$  are two identically distributed independent random variables whose support are included in  $S$ . Due to Lemma 4 (see Appendix) we can write

$$\Pr[SAS_m = SAS_{\hat{m}}] \geq \frac{1}{s}$$

where the equality occurs if and only if the SAS distribution is uniform.

Since  $m$  and  $\hat{m}$  are uniformly distributed in  $\{0, 1\}^t$ , we have  $\Pr[m = \hat{m}] = 2^{-t}$ . Finally, we obtain

$$\Pr[\text{success}] \geq \frac{1}{s} - 2^{-t}$$

where the equality occurs if and only if the SAS distribution is uniform among the set  $S$ .  $\square$

## 4.2 A Generic Multi-Shot Attack

In this section, we sketch a generic attack bounded by  $Q_A$ , resp.  $Q_B$ , instances of Alice, resp. Bob, which works against any message authentication protocol.

**Theorem 2.** *We consider an arbitrary message authentication protocol between Alice and Bob which uses a weak authenticated channel. Let  $S$  be the set of all possible protocol transcripts through the authenticated channel for any input message and let  $s$  be its cardinality.*

*There exists a generic attack which uses  $Q_A$  instances of Alice and  $Q_B$  instances of Bob with probability of success approximatively  $1 - e^{-\frac{Q_A Q_B}{s}}$ .*

*Proof.* We consider that the adversary can use  $Q_A$  instances of Alice and  $Q_B$  instances of Bob. As before, we consider a general man-in-the-middle attack. The adversary first picks  $m_i \in_U \{0, 1\}^t$  with  $i = 1, \dots, Q_A$  and  $\hat{m}_j \in_U \{0, 1\}^t$  with  $j = 1, \dots, Q_B$  and launches several instances of Alice with input  $m_i$ . For each instance of Alice, the attack runs a synchronized protocol with a simulator for Bob and for each instance of Bob, it runs a simulator for Alice with input  $\hat{m}_j$ . Every authenticated message which must be sent by a simulator of Alice is replaced by an authenticated message which has just been received by a simulator of Bob.

Let  $SAS_i$  be the (random) sequence of all authenticated strings (the transcript) which would be exchanged in the protocol between Alice and the simulator of Bob if the simulator were honest, and  $\widehat{SAS}_j$  be the similar sequence between the simulator of Alice and Bob. Let  $S$  the set of all possible protocol authenticated transcripts and  $s$  its cardinality. Note that all  $SAS_i$  and all  $\widehat{SAS}_j$  are independent and identically distributed among the set  $S$  and let  $D$  their distribution.

Clearly, if there exists a couple  $(i, j)$  such that  $\widehat{SAS}_j = SAS_i$ , the attack succeeds. Note that an attack makes sense only if the  $\hat{m}_j$  is different of the  $m_i$ . Consequently, the probability of success can be written as

$$\begin{aligned} \Pr[\text{success}] &= \Pr[\exists i, j \text{ s.t. } (SAS_i = \widehat{SAS}_j \text{ and } m_i \neq \hat{m}_j)] \\ &= 1 - \Pr[\forall i, j, (SAS_i \neq \widehat{SAS}_j \text{ or } m_i = \hat{m}_j)] \\ &\geq 1 - \Pr[\forall i, j, SAS_i \neq \widehat{SAS}_j \text{ or } \forall i, j, m_i = \hat{m}_j] \\ &\geq \Pr[\exists i, j \text{ s.t. } SAS_i = \widehat{SAS}_j] - \Pr[\exists i, j \text{ s.t. } m_i = \hat{m}_j] \end{aligned}$$

$\Pr[\exists i, j \text{ s.t. } m_i = \hat{m}_j]$  is constant for all message authentication protocols and is smaller than  $Q_A Q_B 2^{-t}$ .

Thus, it remains to upper bound  $\Pr[\exists i, j \text{ s.t. } SAS_i = \widehat{SAS}_j]$ . Using Lemma 5 (see Appendix B), we can bound this probability by the one using a uniform distribution, i.e.

$$\Pr[\exists i, j \text{ s.t. } SAS_i = \widehat{SAS}_j] \geq \Pr[\exists i, j \text{ s.t. } SAS_i = \widehat{SAS}_j | D \text{ is uniform}].$$

Consequently, we can approximate the probability of collision using the birthday paradox and finally we obtain

$$\Pr[\text{success}] \approx 1 - e^{-\frac{Q_A Q_B}{s}} - Q_A Q_B 2^{-t}.$$

$\square$

### 4.3 A Generic Multi-Shot Attack Against Non-Interactive Protocols

We can also provide a generic attack against *any* non-interactive message authentication protocol (NIMAP).

**Theorem 3.** *We consider an arbitrary NIMAP between Alice and Bob which uses a weak authenticated channel. Let  $S$  be the set of all possible protocol transcripts through the authentication channel for any input message. Let  $s$  be its cardinality. There exists a generic attack which uses  $Q_A$  instances of Alice, one instance of Bob, and an offline complexity  $\mathcal{O}(T)$  with probability of success approximately  $1 - e^{-\frac{T \cdot Q_A}{s}}$ .*

*Proof.* We consider the generic attack in which the adversary starts by simulating  $T$  Alice instances launched with random inputs  $\hat{m}_i$  and obtains a list of possible SAS, i.e.  $\widehat{SAS}_i$ . Then, he launches  $Q_A$  real instances of Alice with random inputs  $m_j$  and consequently obtains  $Q_A$  authenticated SAS, i.e.  $SAS_j$ . The attack succeeds when at least one authenticated SAS released by Alice corresponds to a computed one, i.e. there exists  $k, \ell$  such that  $SAS_k = \widehat{SAS}_\ell$ . The adversary can launch a single Bob with input  $\hat{m}_\ell$  by simulating Alice and can use  $SAS_k$  for the authentication when needed.

If the distribution of all SAS is uniform, we have a birthday effect and thus the probability of success is approximately  $1 - e^{-\frac{T \cdot Q_A}{s}}$ . When the distribution is not uniform, the probability is even larger due to Lemma 5 (see Appendix B).  $\square$

### 4.4 A Short Overview

In conclusion, we have three theorems which are proving the best expected security of *any* message authentication protocol. More precisely,

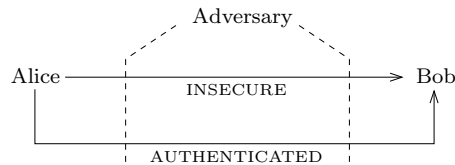
**Theorem 1** says that there exists a one-shot attack against *any* message authentication protocol, either interactive or non-interactive, which succeeds with probability essentially  $\frac{1}{s}$  where  $s$  is the size of the set  $S$  of all possible transcripts.

**Theorem 2** says that there exists a generic attack bounded by  $Q_A$  instances of Alice and  $Q_B$  instances of Bob against *any* message authentication protocol which succeeds with probability essentially  $1 - e^{-\frac{Q_A Q_B}{s}}$ .

**Theorem 3** says that there exists a generic attack bounded by a complexity  $T$  against *any* NIMAP which uses a weak authenticated channel which succeeds with probability essentially  $1 - e^{-\frac{T \cdot Q_A}{s}}$  where  $Q_A$  is the number of instances of Alice used. Hence, they cannot be secure unless  $T \cdot Q_A$  is negligible against  $s$ .

## Non-Interactive Authentication Protocols

We call *non-interactive* message authentication protocols (NIMAP) the message authentication protocols in which the insecure and the authenticated channels are *unidirectional*, i.e. from Alice to Bob. Consequently, Alice can only send messages and Bob can only receive it.



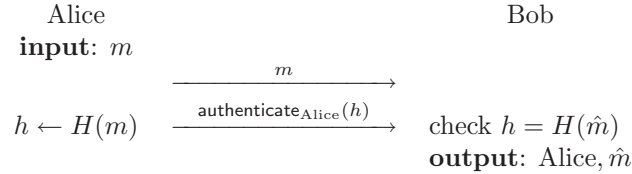
**Figure 5.1.** Non-Interactive Channels Model

We will start to study two popular protocols. The first one is used nowadays in many applications which require key exchange, for instance applications such as GPG or SSH. It was presented by Balfanz et al. [BSSW02] and requires a collision resistant hash function (CRHF) but weak authentication only. To achieve enough security, it requires 160 authenticated bits. The second protocol was published by Gehrman-Mitchell-Nyberg [GMN04]. It allows to authenticate a message with only 16–20 authenticated bits but requires a stronger authenticated channel.

In a second time, we will propose a new protocol which has the same security than the first one presented but using less authenticated bits, without any stronger communication model, and without requiring the hash function to be collision-resistant. Our protocol is based on a trapdoor commitment scheme.

## 5.1 A Non-Interactive Authentication Protocol Based on a Collision-Resistant Hash Function

On Fig. 5.2 is depicted the non-interactive protocol presented by Balfanz et al. [BSSW02] which is based on a collision resistant hash function.



**Figure 5.2.** Non-Interactive Message Authentication using a CRHF.

Note that the authenticated string is fixed by a non-probabilistic algorithm from the input message  $m$ , i.e. the authenticated string is  $H(m)$ . Thus, the authenticated string is the same for each instance of the protocol which uses the same input  $m$ . This particularity allows adversaries to run attacks completely offline. An adversary has *simply* to find a collision on the hash function  $H$  between two messages  $m_1$  and  $m_2$ , i.e.  $H(m_1) = H(m_2)$  and then succeeds with probability 1.

**Theorem 4** ([Vau05]). *Let  $\mu$  be the overall time complexity of the message authentication protocol in Fig. 5.2 using weak authentication. We denote by  $T$ ,  $Q$ , and  $p$  the time complexity, number of oracle queries launch, and probability of success of adversaries, respectively. There is a generic transformation which transforms any adversary into a collision finder on  $H$  whose complexity is  $T + \mu Q$  and probability of success is  $p$ .*

In short, the best known offline attack against this protocol is the collision attack. Using a time complexity of  $T$  hash computations, an adversary has a probability of success of  $1 - e^{-\frac{1}{2}T^2 2^{-k}}$ . It clearly succeeds for  $T = \mathcal{O}(2^{k/2})$ .

The generic attack against non-interactive protocol of Theorem 3 can be run here. Any adversary has a probability of success  $1 - e^{-\frac{T \cdot Q_A}{s}}$  where the adversary is bounded by  $T$  hash computations and  $Q_A$  instances of Alice and a single instance of Bob. Obviously, this attack is not optimal for small  $Q_A$ .

Collision resistance requires the number of authenticated bits to be at least 160 and cannot be reduced considering offline attacks and using only weak authentication.

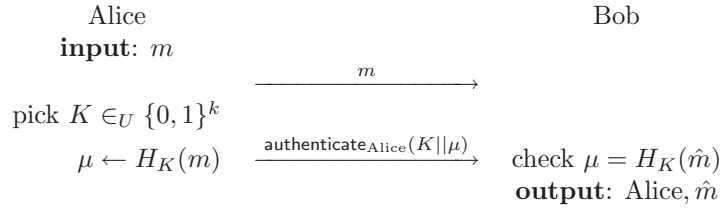
## 5.2 A Non-Interactive Authentication Protocol using Strong Authentication

On Fig. 2.2 is depicted a non-interactive authentication protocol taken from Gehrman-Mitchell-Nyberg [GMN04]<sup>1</sup>. They have proposed three protocols, MANA I, MANA II, and MANA III. The name MANA was chosen for **MAN**ual **A**uthentication.

<sup>1</sup>Note that the original MANA I protocol is followed by an authenticated acknowledgment from Bob to Alice in [GMN04].

In short, MANA I was designed for situation where one device has a display and the other has a keypad. MANA II is a variant of MANA I for situations where the second device has in addition a display. Finally, MANA III was designed for situation where both devices have a keypad. Note that MANA I and MANA II were originally published in [GN01]. Then, the first one was called MANA I in [GN04] and Mechanism I in [MUM03].

MANA I uses a universal hash function family  $H$ . Proposed constructions lead to 16–20 bit long SAS values but require strong authentication<sup>2</sup>. Indeed, using weak authentication, an adversary who gets  $\text{authenticate}(K||\mu)$  has enough time to find a message  $\hat{m}$  such that  $\mu = H_K(\hat{m})$  and to substitute  $m$  with  $\hat{m}$ . We can also achieve security with a stronger authenticated channel which achieves stall-free transmissions.



**Figure 5.3.** The MANA I Protocol.

Using only an authenticated hash value, i.e.  $k = 0$ , an offline attack succeeds with probability 1 and time complexity  $\mathcal{O}(2^{\ell/2})$  by *simply* finding a collision. In fact, it is the protocol of Fig. 5.2. Using only an authenticated random variable makes no sense since it doesn't authenticate the message. MANA I is a trade off: it authenticates the message, but uses a random value to avoid offline attacks.

**Theorem 5.** *Given an  $\epsilon$ -universal hash function family  $H$ , any adversary bounded by a complexity  $T$  and by  $Q_A$  (resp.  $Q_B$ ) instances of Alice (resp. Bob) against the protocol of Fig. 5.3 and using stall-free authentication has a probability of success at most  $Q_A Q_B \epsilon$ .*

*Proof.* A one-shot adversary has no advantage to send  $\hat{m}$  before it has received  $m$  and he cannot send  $\hat{m}$  after  $K||\mu$  is released. Indeed, he would not be able to send  $\hat{m}$  after receiving  $K||\mu$  due to the stall-free assumption. Thus, the attacker must select  $m$  and  $\hat{m}$  and hope that  $H_K(\hat{m}) = H_K(m)$ . Clearly, the assumption on  $H$  limits the probability of success to  $\epsilon$ .

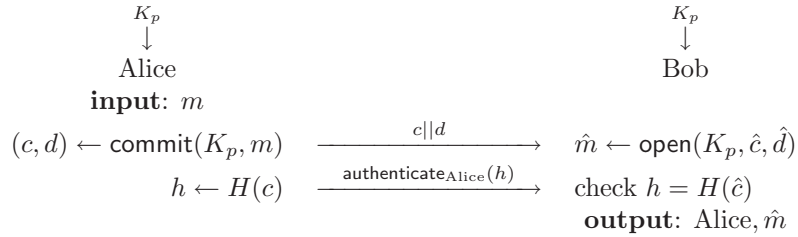
Now, consider powerful adversaries. Using Lemma 1, we can deduce that the probability of success of an adversary is at most  $Q_A Q_B \epsilon$ .  $\square$

We see that any adversary has a non-negligible probability of success when  $Q_A Q_B$  is near  $1/\epsilon$ .

<sup>2</sup>Note that strong authentication renders the protocol “less non-interactive”.

### 5.3 A Proposed Non-Interactive Authentication Protocol using Weak Authentication Only

Consider the protocol depicted on Fig. 5.4 in which the message  $m$  is transmitted by sending  $(c, d) \leftarrow \text{commit}(m)$ . This message can be recovered by anyone using the `open` function. To authenticate this message, the hashed value of  $c$  is sent using an authenticated channel. We prove that this protocol is secure with authenticated strings which can be shorter than in the actual protocol of Fig. 5.2.



**Figure 5.4.** Non-Interactive Message Authentication Based on a WCRHF.

A non-deterministic commitment scheme is the heart of this protocol since an attacker cannot predict the  $c$  value and thus cannot predict the  $H(c)$  value which is the authenticated one. Commitment is assumed to be set up in the Common Reference String (CRS) model.

**Lemma 3.** *Consider the message authentication protocol depicted in Fig. 5.4. We assume that the function  $H$  is a  $(T + \mu, \epsilon_h)$ -weakly collision resistant hash function and the commitment scheme is a  $(T + \mu, \epsilon_c)$ -trapdoor commitment scheme in the CRS model. There exists a (small) constant  $\mu$  such that for any  $T$ , any one-shot adversary against this message authentication protocol with complexity bounded by  $T$  has a probability of success  $p$  smaller than  $\epsilon_h + \epsilon_c$ .*

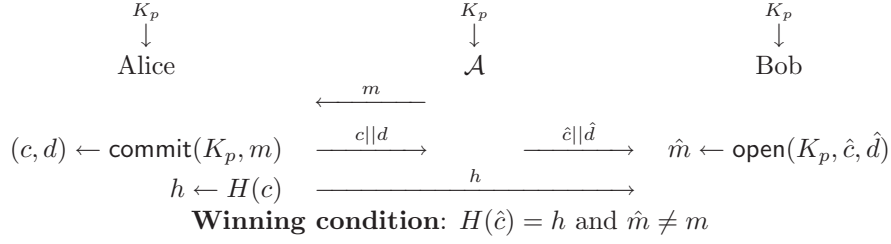
Recall that the  $c$  value is sent through the insecure broadband channel and thus has not to be minimized. Thus, we can use an  $\epsilon_c$  as small as desired since we can use any commitment scheme as secure (as long) as desired.

Assuming that  $H$  is optimally WCR, the best WCR attack using  $T$  hash computations has a probability of success  $\epsilon_h \approx 1 - e^{-T2^{-k}}$ . So, we need  $T = \Omega(2^k)$  to succeed with a one-shot attack. Thus, using the same amount of authenticated bits as the protocol of Fig. 5.2, our protocol has a better resistance against offline attacks. Equivalently, we can achieve the same security as the protocol of Fig. 5.2 using only half amount of authenticated bits, e.g. 80 bits.

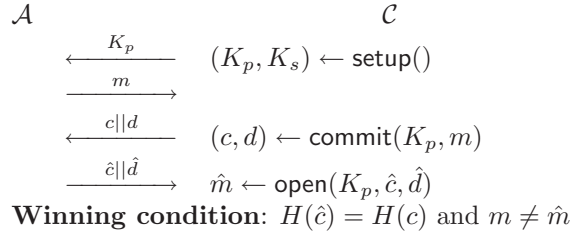
*Proof.* A one-shot adversary  $\mathcal{A}$  against the protocol in Fig. 5.4 follows the game depicted on Fig. 5.5 in which it runs a man-in-the-middle attack.

Clearly, it can be reduced to an adversary  $\mathcal{A}$  who plays the game described in Fig. 5.6.

Assume a one-shot adversary  $\mathcal{A}$  bounded by a complexity  $T$ . Given  $c$ , the adversary  $\mathcal{A}$  has to find a  $\hat{c}$  such that  $H(\hat{c}) = H(c)$ . In addition, it must find a  $\hat{d}$  which opens to  $\hat{m}$  (using



**Figure 5.5.** Game Against the Proposed Protocol.



**Figure 5.6.** Reduced Game.

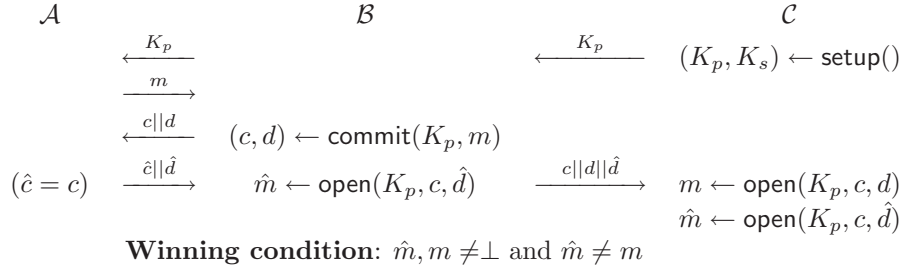
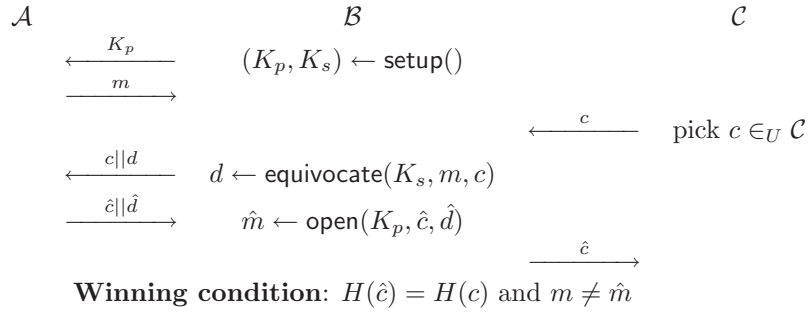
$\hat{c}$ ) which is different from the input  $m$ . He can of course choose a  $\hat{c}$  either equal or either different to  $c$ . We study the two cases.

**Case 1.** ( $\hat{c} = c$ ) The adversary  $\mathcal{A}$  chooses  $\hat{c}$  equal to  $c$  and obviously fulfills the condition  $H(\hat{c}) = H(c)$ . As depicted on Fig. 5.7, we can reduce the adversary  $\mathcal{A}$  to an adversary against the binding game of Fig. 5.4. We use an algorithm  $\mathcal{B}$  bounded by complexity  $\mu$  which plays the binding game with a challenger  $\mathcal{C}$  on one side and simulates a challenger for  $\mathcal{A}$  on the other side at the same time. Using adversary  $\mathcal{A}$  and algorithm  $\mathcal{B}$ , we construct an adversary  $\mathcal{AB}$  which plays the binding game. Note that adversary  $\mathcal{AB}$  has a complexity bounded by  $T + \mu$ .

First, the challenger  $\mathcal{C}$  generates the pair of keys  $(K_p, K_s)$  and sends  $K_p$  to  $\mathcal{B}$ .  $\mathcal{B}$  sends it to  $\mathcal{A}$  and receives a message  $m$  from  $\mathcal{A}$ . He computes  $(c, d)$  using the `commit` function with  $K_p$  and sends  $c||d$  to  $\mathcal{A}$ . As assumed,  $\mathcal{A}$  chooses a  $\hat{c}$  equal to  $c$  and also sends  $\hat{c}||\hat{d}$  to  $\mathcal{B}$ .  $\mathcal{B}$  can now deduce  $\hat{m}$  using the `open` function with inputs  $c$  and  $\hat{d}$ . Finally,  $\mathcal{B}$  sends all required values to the challenger  $\mathcal{C}$ .

Note that  $\mathcal{B}$  simulates perfectly a challenger for  $\mathcal{A}$ . Hence,  $\mathcal{A}$  and  $\mathcal{AB}$  win their respective game at the same time. Consequently, both win with the same probability of success. Recall that the probability of success of an adversary bounded by a complexity  $T + \mu$  against the binding game of Fig. 5.4 is smaller than  $\epsilon_c$  when the commitment scheme is a  $(T + \mu, \epsilon_c)$ -trapdoor commitment. Hence, the probability that  $\mathcal{A}$  succeeds and  $c = \hat{c}$  is at most  $\epsilon_c$ .

**Case 2.** ( $\hat{c} \neq c$ ) The adversary  $\mathcal{A}$  searches a  $\hat{c}$  different from  $c$ . As depicted on Fig. 5.8, we can reduce the adversary  $\mathcal{A}$  to an adversary against a second preimage search game. We use an algorithm  $\mathcal{B}$  bounded by a complexity  $\mu$  with the help of one query to the equivocate oracle.  $\mathcal{B}$  plays the second preimage game with a challenger  $\mathcal{C}$  on one side

**Figure 5.7.** Reduction to the SB Game ( $\hat{c} = c$ ).**Figure 5.8.** Reduction to the WCR Game ( $\hat{c} \neq c$ ).

and simulate a challenger for  $\mathcal{A}$  on the other side at the same time. Using adversary  $\mathcal{A}$  and algorithm  $\mathcal{B}$ , we construct an adversary  $\mathcal{AB}$  which plays the second preimage game with the challenger  $\mathcal{C}$ . Note that adversary  $\mathcal{AB}$  has a complexity bounded by  $T + \mu$ .

First,  $\mathcal{B}$  generates the keys and sends  $K_p$  to  $\mathcal{A}$ .  $\mathcal{B}$  receives a message  $m$  from  $\mathcal{A}$  and receives a challenge  $c$  from  $\mathcal{C}$ .  $\mathcal{B}$  can deduce the decommit value  $d$  by calling the oracle  $\text{equivocate}(m, c)$ . Note that  $c$  has been picked uniformly and consequently the distribution of  $(c, d)$  is the same as if they have been yielded by the commit algorithm. Then,  $\mathcal{B}$  can send  $c||d$  to  $\mathcal{A}$ .  $\mathcal{A}$  sends a  $\hat{c}||\hat{d}$  to  $\mathcal{B}$ . Finally,  $\mathcal{B}$  sends it to the challenger  $\mathcal{C}$ .

Note that  $\mathcal{B}$  simulates perfectly a challenger for  $\mathcal{A}$ . Hence,  $\mathcal{A}$  and  $\mathcal{AB}$  win their respective game at the same time and consequently with the same probability of success. Recall that the probability of success of an adversary against a second preimage game bounded by a complexity  $T + \mu$  is smaller than  $\epsilon_h$  when  $H$  is a  $(T + \mu, \epsilon_h)$ -weakly collision-resistant hash function. Hence, the probability that  $\mathcal{A}$  succeeds and  $c \neq \hat{c}$  is at most  $\epsilon_h$ .

We conclude that any one-shot adversary bounded by a complexity  $T$  against the protocol of Fig. 5.4 has a probability of success smaller than  $\epsilon_c + \epsilon_h$  when the protocol uses a  $(T + \mu, \epsilon_h)$ -weakly collision resistant hash function  $H$  and a  $(T + \mu, \epsilon_c)$ -trapdoor commitment scheme.  $\square$

We now consider powerful adversaries.

**Theorem 6.** *Consider the message authentication protocol depicted in Fig. 5.4. We assume that the function  $H$  is a  $(T + \mu, \epsilon_h)$ -weakly collision resistant hash function and the commit-*

ment scheme is a  $(T + \mu, \epsilon_c)$ -trapdoor commitment scheme in the CRS model. There exists a (small) constant  $\mu$  such that for any  $T$ , any adversary against this message authentication protocol with complexity bounded by  $T$  and with number of Alice's (resp. Bob's) instances bounded by  $Q_A$  (resp.  $Q_B$ ) has a probability of success  $p$  at most  $Q_A(\epsilon_h + \epsilon_c)$ .

Assuming that WCR hash functions and trapdoor commitments such that  $\epsilon_c \ll \epsilon_h = \mathcal{O}(T2^{-k})$  exist, we have  $p = \mathcal{O}(T \cdot Q_A 2^{-k})$  which meets the data of the generic attack from Theorem 3 for  $T \cdot Q_A \ll 2^k$ . Hence, our protocol is essentially optimal. As an example, assuming that an adversary is limited to  $Q_A \leq 2^{10}$ ,  $T \leq 2^{70}$ , and that the security level requires  $p \leq 2^{-20}$ , the protocol of Fig. 5.2 requires  $k \geq 160$  and our protocol requires  $k \geq 100$ . Using MD5 [Riv92], our protocol still achieves a quite luxurious security even though collisions have been found on MD5 [WY05].

*Proof.* Consider an adversary who launches  $Q_A$  instances of Alice and  $Q_B$  instances of Bob. Clearly, we can simulate all instances of Bob, pick one who will make the attack succeed, and launch only this one. Hence, we reduce to  $Q_B = 1$ . Recall from Lemma 3 that any one-shot adversary has a probability of success smaller than  $\epsilon_h + \epsilon_c$ . Using Lemma 1, we conclude that any adversary has a probability of success at most  $Q_A(\epsilon_h + \epsilon_c)$ .  $\square$

In conclusion, using the same amount of authenticated bits than the protocol of Fig. 5.2, we conclude that both have the same security against online attacks, but this latter protocol has a better resistance against offline attacks.

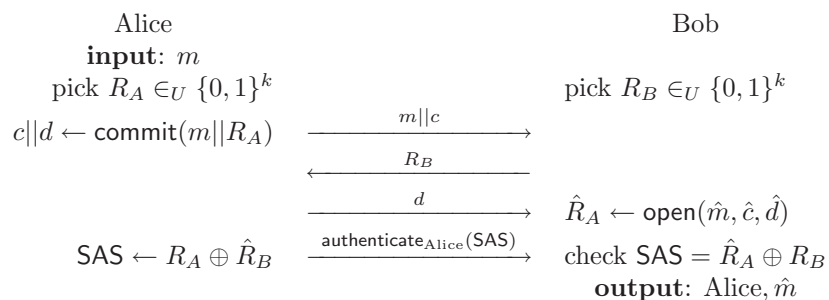


# Interactive Authentication Protocol

In this chapter, we will recall the Vaudenay [Vau05] SAS-based message authentication protocol. In a second time, we will present two implementations. The first one is a modification of OpenSSH which allows users to use this protocol. The second application is a generic peer-to-peer file authentication which allows two distant users to authenticate any files such as public keys, and if required to exchange it assuming that they know the distant voice and they can authenticate it.

## 6.1 The Vaudenay SAS-based Protocol

On Fig. 6.1 is depicted the SAS-based message authentication protocol taken from Vaudenay [Vau05]. In this protocol, Bob will authenticate a message  $m$  from Alice. After the exchange of the three first messages through the insecure channel, Alice has to authenticate a short string  $SAS = R_A \oplus R_B$  where  $R_A$  and  $R_B$  are randomly selected at the beginning by Alice and Bob respectively.



**Figure 6.1.** SAS-based Message Authentication.

Recall that using a weak authentication channel, the adversary can delay or replay au-

thenticated messages. These two channel defects can lead to powerful attacks as seen on the beginning of Chapter 4. This protocol resists to general attacks run by any adversary bounded by a number of protocol runs, i.e. the online complexity. Indeed, assuming that hiding commitment schemes exist, this protocol resists to offline attacks.

## 6.2 Short Security Analysis, Optimality

Due to interactivity, this protocol does not allow offline attacks. In addition, the authenticated string is uniformly distributed and independent of the message  $m$ . Vaudenay [Vau05] proves that a one-shot attack has a probability of success at most  $2^{-k} + \epsilon$  where  $\epsilon \geq 0$  represents a small advantage for attackers due to a non-perfect commitment scheme (non perfectly hiding).

**Theorem 7** ([Vau05]). *We consider one-shot adversaries against the message authentication protocol in Fig. 6.1. We denote by  $T$  and  $p$  their time complexity and probability of success, respectively. We assume that the commitment scheme is either  $(T_C, \epsilon)$ -extractable or  $(T_C, \epsilon)$ -equivocable. There exists a (small) constant  $\mu$  such that for any adversary, we have either  $p \leq 2^{-k} + \epsilon$  or  $T \geq T_C + \mu$ .*

In addition, he proves in [Vau05] that attacks which use  $Q_A$  instances of Alice and  $Q_B$  instances of Bob have a probability of success against this protocol at most  $Q_A Q_B (2^{-k} + \epsilon)$ . Note that the proof was done using Lemma 1. Consequently, any adversary which can use at least  $Q$  instances of Alice and Bob have a probability of success less than  $\frac{Q^2}{4} (2^{-k} + \epsilon)$ . This meets the data of the generic attack from Theorem 1 for  $q \ll 2^k$ . So, the protocol is essentially optimal.

Theorem 1 tells us that the security of the protocol of Fig. 6.1 is optimal among all comparable protocols since it reaches the maximal security of a message authentication protocol and these it cannot exist a better protocol using the same amount of authenticated bits.

## 6.3 A First Application: OpenSSH

Since SSH uses public key cryptography, it needs public key exchange and thus an authentication step is required. Most of the time, the authentication of the distant public key is made by comparing the footprint computed locally by the client with the distant footprint, which is obtained from the SSH server in an authenticated way. One advantage of this way is the non-interactivity. Fig. 6.2 shows a possible SSH connection setup between a client and a server.

Actually, the *major drawback* is that users often accept the distant public key by just answering “yes” to the first question without checking the fingerprints. This leads to the non-authentication of the distant public key and thus to a security problem.

To improve the security of OpenSSH, we have implemented the Vaudenay SAS-based message authentication protocol [Vau05] in addition to the existent method. We use the authentication protocol to authenticate the remote public key (from the server). This new method forces the user to authenticate the public key since it can not continue until it has not entered the valid SAS. Indeed, he must type the true SAS (and not just answer “yes”).

```

#> ssh pasini@lasecpc12
The authenticity of host 'lasecpc12 (128.178.73.67)' can't be
established.
RSA key fingerprint is
fb:0c:0e:bc:fd:c7:3d:68:e9:8b:2d:6b:c6:ae:88:e2.
Are you sure you want to continue connecting (yes/no)? yes

30651: Warning: Permanently added 'lasecpc12,128.178.73.67' (RSA)
to the list of known hosts.

Password: *****

Have a lot of fun...

#> exit
logout
Connection to lasecpc12 closed.

```

**Figure 6.2.** Connection to a Remote Host using a Conventional Protocol

Unconscious users cannot use the system as before accepting all keys without authentication. Thus, this new system guarantees to the network a higher security. Fig. 6.3 gives us an example of an OpenSSH connection setup using the Vaudenay SAS-based message authentication.

We have tried to render our implementation compatible with other OpenSSH systems. Indeed, we have added the option **-Z** to both client and server. In particular, if the server would accept to use the Vaudenay SAS-based protocol [Vau05], the administrator must specify the option **-Z** when he launches the daemon. Consequently, when the server send his version to the client, the string is **SSH – 1.99 – OpenSSH\_3.9p1-sas**. The last option indicates that the server could use the Vaudenay SAS-based protocol [Vau05]. The conventional implementations of OpenSSH would accept this version since it stops before the **-**. Similarly, if the client would use the Vaudenay SAS-based protocol [Vau05] for the authentication, he have to specify the option **-Z** in the command line. During the parameters negotiation, they determine whether they use the new protocol or not. Four cases are now possible. The new protocol is of course used only if both have enabled the SAS protocol, i.e. they sends a version terminating with **-sas**.

Recall that the only manner to attack the SAS-based protocol of Fig. 6.1 is launching several protocol instances which is not hard to detect.

This method of authentication *would be perfect* if it did not have the following defect:

The SAS must be obtained (and authenticated) for each unknown distant public key, for instance by phone if the administrator voice is known, or physically otherwise.

Suppose an administrator is responsible of a very big number of hosts, for instance a department of EPFL. He has to respond to a big number of phone calls or to encounter some students since his voice is not known by them. When a user starts a connection with an unknown host, he needs the corresponding SAS. The administrator must be available at this moment

On the client side:

```
#> /scratch/pasini/myssh/bin/ssh -Z pasini@lasecpc16
From the server : SSH-1.99-OpenSSH_3.9p1-sas
Use SAS authentication.
Start SAS protocol:
  Wait on commit... Done.
    Commit=22348769876106551
  Pick Rc. Rc=747
  Send Rc.
  Wait on decommit...Done.
    Fingerprint=37:3a:3d:96:4b:c8:04:de:bc:3a:2b:c3:5c:d5:bd:f6
    Rs=990
  Commit received valid

  Enter the SAS: 1737 [entered by the user]
  SAS correct

pasini@lasecpc16's password: ***** [entered by the user]

#> exit
logout
Connection to lasecpc16 closed.
```

On the server side (daemon output):

```
#> /scratch/pasini/myssh/sbin/sshd -Z
SAS authentication enabled by the daemon.
Start infinite loop

Child created to handle connection, pid=28649.
Start SAS protocol:
  Compute commit and decommit
    Decomit=37:3a:3d:96:4b:c8:04:de:bc:3a:2b:c3:5c:d5:bd:f6-990
    Commit=22348769876106551
  Send the commit.
  Wait on Rc...Done.
    Received Rc=747
  Send the decommit.
  Compute the SAS.
    SAS=1737
  Save the SAS in the SAS database (for administrator).
```

On the server side (SAS database):

```
128.178.73.71 1737 [new line created]
```

**Figure 6.3.** Connection to a Remote Host using the SAS-Based Protocol

which can be impossible in some cases. Using the conventional way, the administrator must be available too, but not necessarily at the same time since the protocol is non-interactive. In conclusion, this new system is more secure in particular for unconscious users. On the other hand, it is not practically usable in most cases.

Possible applications of this implementation can be the exchange of public keys between distant users (or companies) that require a high security and very quickly. As presented in the introduction, suppose for instance a disaster cause in a bank and the administrator is in trip. Thus, he must establish a very quick connection with the server using a new generated pair of keys. The administrator is forced to authenticate the public key (very) quickly. Using the SAS-based protocol, he has just to call the bank, starts the protocol, and finally types the SAS which has been obtained with the help of a responsible person. In this case, the security can be achieved with few bits, typically 15 bits, which is much less than the 160 bits of a fingerprint.

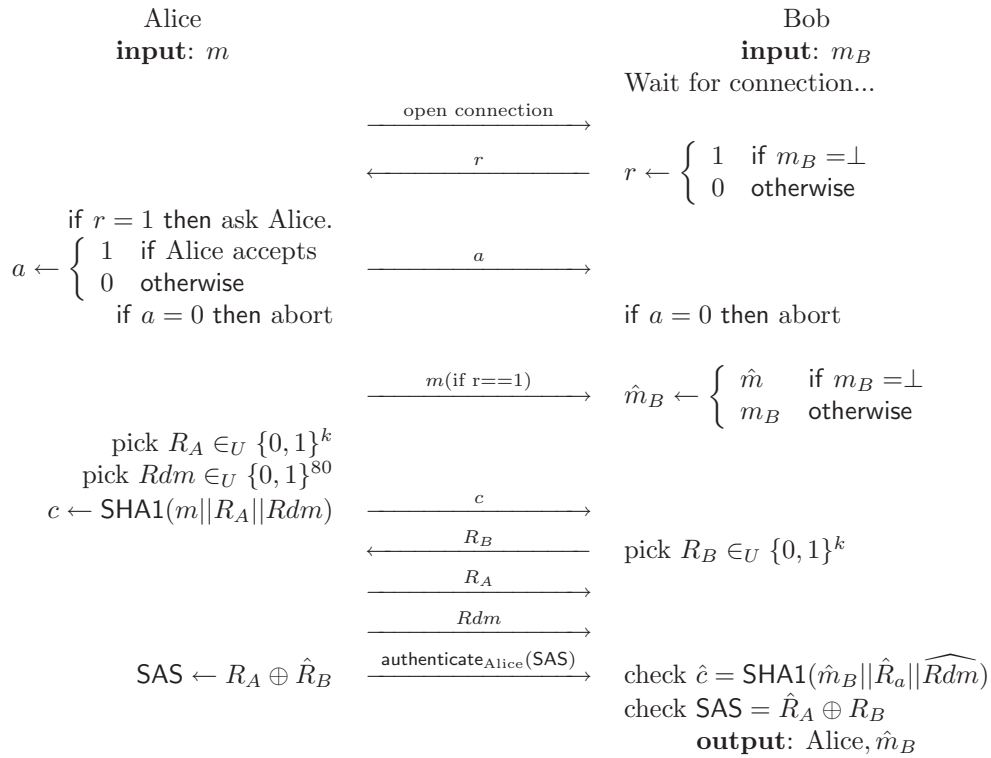
## 6.4 A Peer-To-Peer File Authentication

In Section 6.3, we have seen that the implementation in OpenSSH of the Vaudenay SAS-based message authentication protocol [Vau05] was a problem. Indeed, the interactivity forced the administrator to be available at “any time” to exchange some SAS. This reason encouraged us to propose a better application for this protocol. It seems reasonable to assume that two users, who can arrange themselves between them, can be available during the exchange of a message, e.g. a public key. In this section, we propose a peer-to-peer application that help to authenticate public keys. In fact, the proposed system is more general and allows users to authenticate, and if required to exchange, files like GPG public key files.

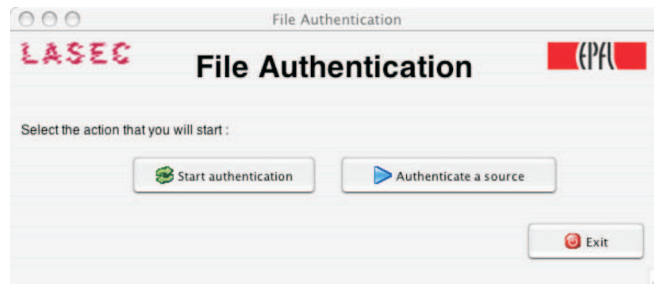
The application which have been implemented communicates through the Internet network using a chosen port (the insecure channel). In addition, they needs an human communication channel, which achieves authentication, typically telephone (the authenticated channel). The application is based on the client-server model and works in short as follow: Suppose two persons: Alice and Bob. Bob would authenticate a file from Alice (e.g. her GPG public key). First, he launches the SAS File Exchange application and wait for connection (i.e. he is the server). Then, he contacts Alice, for example by telephone, and asks him to authenticate one of her files. Alice launches the application too, selects the file to authenticate and the destination address, i.e. IP address and port number. Then, she starts the authentication protocol (i.e. she is the client).

### 6.4.1 The Implemented Protocol

The Vaudenay SAS-based authentication protocol depicted on Fig. 6.1 has been adapted for a practical implementation. The implemented version works as described on Fig. 6.4. In particular, the file which has to be authenticated can be exchanged or not. If Alice and Bob has already exchanged the file, e.g. by email, authentication only is made, otherwise the file is exchanged at the beginning of the protocol.



**Figure 6.4.** Implementation of the Vaudenay SAS-based Message Authentication with Random Oracle Commitment.



**Figure 6.5.** Screen-shot of the Main Frame

### 6.4.2 The Final Application

The final application has been implemented in Java for its simplicity of designing user-friendly interfaces. It is principally composed of three windows.

**The main window** is the only visible at the launches of the application. It allows users to open one of the two other windows or to quit the application. A screen-shot is printed on Fig. 6.5.

**The receive window** is normally opened by the user, i.e. Bob, who will authenticate a distant file, i.e. from Alice, and possibly will receive it. A screen-shot is printed on Fig. 6.7.

He has to specify the port number and the file destination. Three cases are possible for the file destination:

**No filename specified.** The protocol (Bob) requests the source (Alice) to send the file. If the source accept, Bob receives the file and the correspondent filename. The received file would be saved in the received filename.

**A non-existent filename specified.** The protocol (Bob) requests the source (Alice) to send the file. The received file would be saved in the specified filename.

**An existent filename specified.** The protocol (Bob) does not request the source (Alice) to send the file and later use the specified file to check the commit.

By clicking on the button “Waiting authentication”, Bob enters in a waiting mode until it receives a connection on the specified port.

After the execution of the protocol, Bob has to enter the SAS which has been yield by Alice for instance by telephone.

**The send window** is normally opened by the user who has the source of the file, i.e. Alice. A screen-shot is printed on Fig. 6.6.

She has to specify the source filename, the destination hostname, i.e. the on of Bob, and the port number to be used.

By clicking on the button “Start authentication”, Alice starts the protocol by establishing a connection to Bob.

If Bob would have requested the file transfer, a dialog window is opened. Alice can either accept to send the file, or either abort the protocol.

At the end of the execution of the protocol, Alice has to give the SAS to Bob using an authenticated channel, for instance a telephone.

The executions of the SAS-based protocol on Alice and on Bob sides are depicted respectively on Fig. 6.6 and Fig. 6.7. These executions represent a public key authentication without transfer by considering that Alice has sent its public key to Bob prior the protocol, e.g. by email.

### 6.4.3 Number of SAS Trials

The used SAS has a fixed length of 5 digits. A sixth digit has been added to check its validity. In an application of this type, it is important to allow users to make mistake(s). On the other hand, this flexibility does not reduce the security. If simply two tries are allowed, an attacker has two tries to attack the protocol. Consequently, the probability of success is twice the probability of success of the original protocol.

The sixth digit is a digit of redundancy. It is the sum of the five original digits modulo 10. When Bob enters a SAS, first the redundancy is checked.

**If the redundancy is bad**, Bob has entered a non-valid SAS and can try another one (only). Note that an attacker has no chance with a generated SAS with bad redundancy since it would be rejected with probability 1.

**If the redundancy is correct**, either the SAS entered is right and also the file is authenticated, either it is false and the file authentication is aborted since an attack can be occurred.

Using this method, users can make a mistake typing the SAS, but the security is not decreased. On the other hand, it costs one additional digit.

### 6.4.4 In Short

The proposed application allows users to exchange files on an authenticated way. It can be used to exchange PGP, or GPG, public keys by authenticating only six digits. Note that the current method uses the protocol of Fig. 5.2 and requires 160 authenticated bits which are usually represented by 32 hexadecimal digits. Fortunately, the proposed protocol requires the exchange of only 15 authenticated bits which is much smaller than the current method.



Figure 6.6. Screen-shot of the Frame on Alice Side (Verbose Mode)

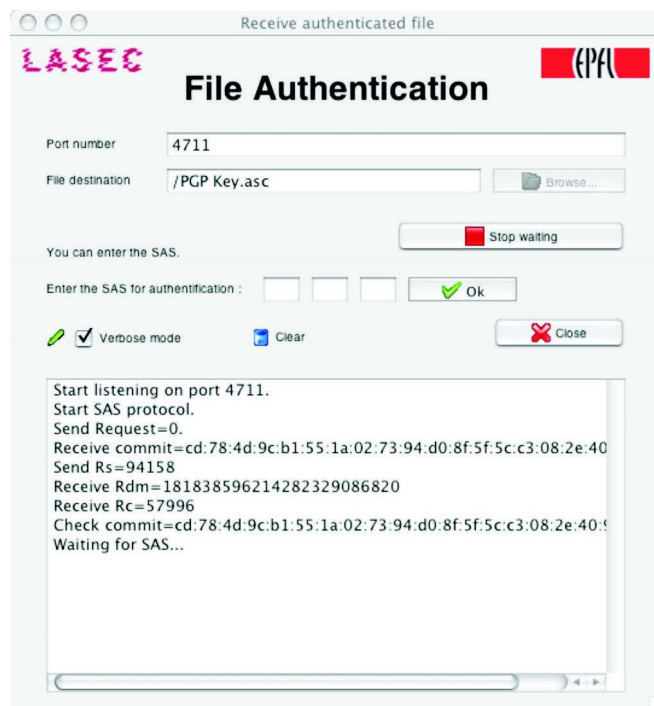


Figure 6.7. Screen-shot of the Frame on Bob Side (Verbose Mode)



# Interactive vs. Non-Interactive Protocols and their Applications

In Chapter 6, we have seen two applications of the interactive SAS-based protocol. In the OpenSSH application, the interactivity was a problem, recall that the administrator became quickly overloaded by the SAS distribution. In the second application, the peer-to-peer file authentication, the interactivity was an advantage since it allowed users to exchange short strings instead of longer ones.

In this chapter, we try to compare and analyze interactive and non-interactive protocols. In a second time, we sketch some applications and we discuss whether interactive or non-interactive is better in this case.

## 7.1 A Short Comparison

**Usability** An interactive protocol allows shorter authenticated strings and thus it is more user-friendly. On the other hand, an interactive protocol forces the users to be synchronized which can be not user-friendly.

**Security** We have seen in Chapter 5 that non-interactive protocols are vulnerable to offline and online attacks. Interactive protocols are in general resistant to offline attacks and thus their security is better.

**Cost** In term of amount of authenticated bits, interactive protocols are lower cost since they don't have to resist to offline attacks and thus can use shorter authenticated string

An exception is MANA. Indeed, it is a non-interactive protocol using only few authenticated bits. On the other hand, it requires a stronger authenticated channel which is more expensive than a weak authenticated one.

**Complexity** The non-interactive protocols often use one-way hash functions. Thus, their complexity is basically a hash computation on each sides of the protocol.

In the proposed (non-interactive) protocol, a commitment scheme is used in addition to the hash function. Consequently, its complexity is a commitment and a hash computation on each side.

The only interactive protocol seen in this work uses a commitment and its complexity is a commit call and an open call on each side respectively.

Note that a commitment scheme can be very complex compared to a simple hash computation. Thus, protocols using a commitment scheme are in general more complex than ones using only hash functions.

## 7.2 Some Applications Examples

### 7.2.1 Applications Based on Public Keys

**SSH.** In Section 6.3, we have seen the problem of an interactive protocol in this case. On the other hand, the interactivity forces users to authenticate the distant public key and not just accept them. Indeed, with the presented solution, the user must enter the good SAS to continue.

**PGP, GPG.** Currently, the protocol used is non-interactive. Consequently, it allows the two distant users to run a non-synchronized authentication. On the other hand, they must authenticate a long string, e.g. 160 bits.

In Section 6.4, we have seen that an interactive protocol can be used for peer-to-peer key (file) authentication. Here, we assume that two distant users can run a synchronized protocol. The interactivity has the advantage of allowing shorter SAS.

**PGPfone.** PGPfone is a software package that allows to transform a computer into a secure telephone. Suppose two persons would communicate securely, i.e. Bob calls Alice. A possible scenario can be to start with a non-secure communication. The two users can communicate. In particular, if Bob knows Alice, the non-secure communication is an authenticated channel. Thus, Alice can send its public key to Bob, e.g. using another TCP port or by email, and she can authenticate their public key by spelling its fingerprint. After this authentication step, the two users can setup a secure communication by using the public key which has just been exchanged.

Clearly, both users are eventually synchronized, thus an interactive protocol can be run simultaneously on another TCP port without any additional constraint. In addition, this second method allows shorter SAS as seen in the comparison since the protocol is interactive.

In conclusion, an interactive protocol is better in this case since it allows shorter SAS without requiring any additional constraint.

### 7.2.2 Bluetooth Devices Pairing and Wireless USB

An alternative method to Bluetooth pairing can be to exchange a public key and then authenticate it. Note that a device pairing requires both available. Consequently, an interactive protocol requires no additional constraint. Thus, short SAS can be used to authenticate public keys. For small devices, this can be restrictive since costly commitment schemes can be hard to implement. Recall that protocol of Fig. 6.1 requires a commitment scheme.

Wireless USB will appear soon. Some USB devices, such as printers, will become wireless. As for Bluetooth, a WUSB device requires a pairing step otherwise attacks can be run.

Let us consider some applications. We do not focus on the implementation of the authentication protocol, but rather on the user interface.

A headset has no keyboard and no display and it must communicate to exchange the SAS (if an interactive protocol is used). Clearly, a telephone must authenticate its headset. After the protocol execution, a SAS must be exchanged from the headset to the telephone. A solution would be that the ear-phone pronounces the SAS and then the user types it on the telephone keyboard.

For a keyboard of a computer, it is important to authenticate the computer and to setup a secure communication. Otherwise, an adversary can for example recover passwords typed on the keyboard. Thus, a SAS must be exchanged from the computer to the keyboard. Fortunately, the keyboard has inputs. The users could type on his keyboard a SAS which is displayed on the screen of the computer. Here, we consider that the display to user is an authenticated channel. Note that it does not need a confidential channel since a SAS is not a PIN code. A symmetric problem is to make sure that the computer is linked to the right keyboard. For this, we only require the keyboard to have a LED showing when the secure communication was successfully set up: if the computer says that a connection was setup to a keyboard but the right keyboard does not, then it was made by a wrong one.

When a user on the road would like to print a confidential document from his laptop computer to a wireless printer, authentication of the printer is necessary. In the case where the printer has a display, it is not a problem: the printer just displays the SAS which is typed on the laptop. Consider that the printer has no display. The printer can still print the SAS on a page. The protocol can be interactive in this case to allow short SAS.

A hard disk has in general no input/output with the user. Here are some alternate ways for “deaf-mute” devices:

1. Start with a wired connection to setup a security association. After this step, it can be disconnected and can use the wireless link securely. Note that a good solution can be that the device remembers the security association for the next time.
2. “Hard-code” a key on the device and print them on its box. The user has to type it on its computer during the authentication step.
3. Use *resurrecting duckling* paradigm from Stajano-Anderson [SA99]. In short, when a device is booted the first time, it searches another device in its vicinity and trusts the first found for always.
4. Use a distance bounding authentication which is well adapted in these cases since we are doing pairings and devices are very close.



# Conclusion

We have analyzed on a global perspective the security of message authentication protocols. In particular, we have seen that there exists a generic one-shot attack with probability of success essentially  $1/s$  where  $s$  is the size of the set of all possible authenticated strings. We have also sketched a generic attack which uses  $Q_A$  instances of Alice and  $Q_B$  of Bob with probability of success essentially  $1 - e^{-\frac{Q_A Q_B}{s}}$ . Thus, any message authentication protocol which has no better attack is essentially “optimal” since any other protocol can be attacked using these generic attacks and consequently can not have a better security.

We have discussed the security of the non-interactive protocols of Fig. 5.2 and 5.3. We have proposed a new non-interactive message authentication protocol based on a commitment scheme. It has the same security as the currently used in SSH (Fig. 5.2) against one-shot attacks but using only half authenticated bits, e.g. 80 bits. 100 bits only are required against more general attacks and allowing a quite good security. Indeed, due to the commitment scheme, the authenticated value is not foreseeable and the proposed protocol is resistant to collision attacks that are run offline. We stress that the security of our protocol relies essentially on the hardness of the SB game of the commitment scheme and on the hardness on the WCR game of the hash function.

In addition, we have proposed two implementations of the Vaudenay SAS-based authentication protocol. The first was an implementation in OpenSSH and the second was a peer-to-peer file authentication. These two implementations brought us to discuss whether an application must use an interactive protocol or if it is better to use a non-interactive one. As expected, it is in fact dependent on the application. Indeed, an interactive protocol allows shorter SAS than a non-interactive one, but requires a synchronization between the users. For instance, an interactive protocol in SSH is not well adapted since the synchronization leads to some problems, but for the second application an interactive protocol is better since it allows shorter authenticated strings and thus becomes more user-friendly.



# Bibliography

- [AES01] Advanced encryption standard (AES). Federal Information Processing Standards, Publication 197, U.S. Department of Commerce, National Institute of Standards and Technology, 2001.
- [BB05] Laurent Bussard and Walid Bagga. Distance-bounding proof of knowledge to avoid real-time attacks. In *SEC '05: 20th IFIP International Information Security Conference*, Makuhari-Messe, Chiba, Japan, May 2005.
- [BC93] Stefan Brands and David Chaum. Distance-bounding protocols. In Tor Helleseth, editor, *Advances in cryptology – EUROCRYPT '93: Workshop on the Theory and Application of Cryptographic Techniques*, volume 765 of *Lecture Notes in Computer Science*, pages 344–359, Lofthus, Norway, May 1993. Springer-Verlag.
- [BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156–189, 1988.
- [BCJ<sup>+</sup>05] Eli Biham, Rafi Chen, Antoine Joux, Patrick Carribault, Christophe Lemuet, and William Jalby. Collisions of SHA-0 and reduced SHA-1. In *Advances in Cryptology – EUROCRYPT '05: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, *Lecture Notes in Computer Science*, pages 36–57, Aarhus, Denmark, 2005. Springer-Verlag.
- [BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO '96: 16th Annual International Cryptology Conference*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15, Santa Barbara, California, U.S.A., August 1996. Springer-Verlag.
- [BCP03] Emmanuel Bresson, Dario Catalano, and David Pointcheval. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. In Chi-Sung Lai, editor, *Advances in cryptology – ASIACRYPT '03: 9th International Conference on the Theory and Application of Cryptology and Information Security*, volume 2894 of *Lecture Notes in Computer Science*, pages 37–54, Taipei, Taiwan, January 2003. Springer-Verlag.

- [BK90] Joan F. Boyar and Stuart A. Kurtz. A discrete logarithm implementation of perfect zero-knowledge blobs. *Journal of Cryptology*, 2(2):63–76, 1990.
- [Blu03] Bluetooth. Specification of the Bluetooth System. Version 1.2, 2003.
- [BR93] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO '93: 13th Annual International Cryptology Conference*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249, Santa Barbara, California, U.S.A., August 1993. Springer-Verlag.
- [BR00] John Black and Phillip Rogaway. CBC MACs for arbitrary-length messages: The three-key constructions. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO '00: 20th Annual International Cryptology Conference*, volume 1880 of *Lecture Notes in Computer Science*, pages 197–215, Santa Barbara, California, U.S.A., 2000. Springer-Verlag.
- [BSSW02] Dirk Balfanz, Diana K. Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proceedings of Network and Distributed System Security Symposium 2002 (NDSS'02)*, San Diego, California, U.S.A, February 2002.
- [CCH05] Mario Cagalj, Srdjan Capkun, and Jean-Pierre Hubaux. Key agreement in peer-to-peer wireless networks. In *Proceedings of the IEEE, Special Issue in Security and Cryptography*, 2005.
- [CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO '01: 21st Annual International Cryptology Conference*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40, Santa Barbara, California, U.S.A., August 2001. Springer-Verlag.
- [CGHGN01] Dario Catalano, Rosario Gennaro, Nick Howgrave-Graham, and Phong Q. Nguyen. Paillier's cryptosystem revisited. In *CCS '01: Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 206–214, Philadelphia, Pennsylvania, U.S.A., 2001. ACM Press.
- [CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multiparty secure computation. In *STOC '02: Proceedings of the thirty-fourth annual ACM symposium on Theory of computing*, Montreal, Quebec, Canada, 2002. ACM Press.
- [CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT '02: International Conference on the Theory and Applications of Cryptographic Techniques*, volume 2332 of *Lecture Notes in Computer Science*, Amsterdam, The Netherlands, April 2002. Springer-Verlag.

- 
- [DES77] Announcing the Data Encryption Standard (DES). Federal Information Processing Standard, Publication 46, National Institute of Standards and Technology (NIST), 1977.
- [DES99] Data Encryption Standard (DES). Federal Information Processing Standard, Publication 46-3, National Institute of Standards and Technology (NIST), 1999.
- [DG03] Ivan Damgård and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 426–437, San Diego, California, U.S.A., 2003. ACM Press.
- [DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, November 1976.
- [DN02] Ivan Damgård and Jesper Buus Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In Moti Yung, editor, *Advances in Cryptology – CRYPTO '02: 22nd Annual International Cryptology Conference*, volume 2442 of *Lecture Notes in Computer Science*, pages 581–596, Santa Barbara, California, U.S.A., August 2002. Springer-Verlag.
- [DSS94] Digital signature standard (DSS). Federal Information Processing Standard, Publication 186, U.S. Department of Commerce, National Institute of Standards and Technology, 1994.
- [DSS00] Digital signature standard (DSS). Federal Information Processing Standard, Publication 186-2, U.S. Department of Commerce, National Institute of Standards and Technology, 2000.
- [GMN04] Christian Gehrman, Chris J. Mitchell, and Kaisa Nyberg. Manual authentication for wireless devices. *RSA Cryptobytes*, 7(1):29–37, January 2004.
- [GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.
- [GN01] Christian Gehrman and Kaisa Nyberg. Enhancements to Bluetooth baseband security. In *Nordsec '01*, Copenhagen, Denmark, November 2001.
- [GN04] Christian Gehrman and Kaisa Nyberg. Security in personal area networks. *Security for Mobility*, pages 191–230, 2004.
- [IK03] Tetsu Iwata and Kaoru Kurosawa. Omac: One-key cbc mac. In Thomas Johansson, editor, *FSE '03: Fast Software Encryption: 10th International Workshop*, volume 2887 of *Lecture Notes in Computer Science*, pages 129–153, Lund, Sweden, January 2003. Springer-Verlag.
- [JV03] Pascal Junod and Serge Vaudenay. Fox specifications version 1.1. In *Technical Report EPFL/IC/2003/82*, EPFL, 2003.

- [Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.
- [MUM03] International Organization for Standardization, Geneva, Switzerland. ISO/IEC 1st CD 9798-6, Information technology – Security techniques – Entity authentication – Part 6: Mechanisms using manual data transfer, 2003.
- [MVOV96] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
- [MY04] Philip MacKenzie and Ke Yang. On simulation-sound trapdoor commitments. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT '04 : International Conference on the Theory and Applications of Cryptographic Techniques*, volume 3027 of *Lecture Notes in Computer Science*, pages 382–400, Interlaken, Switzerland, May 2004. Springer-Verlag.
- [Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT '99: International Conference on the Theory and Application of Cryptographic Techniques*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 1999. Springer.
- [Pas03] Rafael Pass. On deniability in the common reference string and random oracle model. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO '03: The 23rd Annual International Cryptology Conference*, volume 2729 of *Lecture Notes in Computer Science*, pages 316–337, Santa Barbara, California, U.S.A., 2003. Springer-Verlag.
- [Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO '91: The 11th Annual International Cryptology Conference*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140, Santa Barbara, California, U.S.A., 1991. Springer-Verlag.
- [Riv92] Ronald L. Rivest. The MD5 message digest algorithm. Technical Report Internet RFC-1321,IETF, 1992.
- [RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, Februar 1978.
- [SA99] Frank Stajano and Ross Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, *Security Protocols, 7th International Workshop Proceedings*, volume 1796 of *Lecture Notes in Computer Science*, pages 172–194, Cambridge, UK, 1999. Springer-Verlag.
- [Sha49] C. E. Shannon. Communication theory of secrecy systems. *Bell Sys. Tech. J.*, 28:657–715, 1949.

- 
- [SHA93] Secure hash standard. Federal Information Processing Standard, Publication 180, U.S. Department of Commerce, National Institute of Standards and Technology, 1993.
- [SHA95] Secure hash standard. Federal Information Processing Standard, Publication 180-1, U.S. Department of Commerce, National Institute of Standards and Technology, 1995.
- [Sim88] Gustavus J. Simmons. How to (really) share a secret. In Shafi Goldwasser, editor, *Advances in Cryptology – CRYPTO '88: The 8th Annual International Cryptology Conference*, volume 403 of *Lecture Notes in Computer Science*, pages 390–448, Santa Barbara, California, U.S.A., 1988. Springer-Verlag.
- [Sti95] Douglas R. Stinson. *Cryptography: Theory and Practice*. CRC Press, 1995.
- [Vau04] Serge Vaudenay. *Communication Security : An Introduction to Cryptography*. EFPL Course, 2004.
- [Vau05] Serge Vaudenay. Secure communications over insecure channels based on short authenticated strings. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO '05: The 25th Annual International Cryptology Conference*, volume 3621 of *Lecture Notes in Computer Science*, pages 309–326, Santa Barbara, California, U.S.A., August 2005. Springer-Verlag.
- [WBD05] Yongdong Wu, Feng Bao, and Robert H. Deng. Secure human communications based on biometrics signals. In *SEC '05: 20th IFIP International Information Security Conference*, pages 205–221, Chiba, Japan, May 2005.
- [WLF<sup>+</sup>05] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the hash functions MD4 and RIPEMD. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT '05: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, *Lecture Notes in Computer Science*, pages 1–18, Aarhus, Denmark, 2005. Springer-Verlag.
- [WY05] Xiaoyun Wang and Hongbo Yu. How to break MD5 and other hash functions. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT '05: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, *Lecture Notes in Computer Science*, pages 19–35, Aarhus, Denmark, 2005. Springer-Verlag.
- [WYY05a] Xiaoyun Wang, Yiqun Yin, and Hongbo Yu. Finding collisions in the full SHA1. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO '05: The 25th Annual International Cryptology Conference*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36, Santa Barbara, California, U.S.A., August 2005. Springer-Verlag.
- [WYY05b] Xiaoyun Wang, Xiuyuan Yu, and L. Y. Yin. Efficient collision search attacks on SHA-0. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO '05:*

*The 25th Annual International Cryptology Conference*, volume 3621 of *Lecture Notes in Computer Science*, pages 1–16, Santa Barbara, California, U.S.A., August 2005. Springer-Verlag.

# List of Figures

1.1	The Common Human Communications Channels . . . . .	15
1.2	The Shannon Model [Sha49]. . . . .	16
1.3	The Merkle-Diffie-Hellman Model. . . . .	16
1.4	The Diffie-Hellman Key Agreement Protocol. . . . .	17
1.5	The Semi-Authenticated Key Transfer Using Public-Key Cryptography. . . . .	17
2.1	Non-Interactive Message Authentication based on a CRHF . . . . .	20
2.2	The MANA I Protocol. . . . .	21
2.3	Interactive SAS-based Message Authentication. . . . .	21
2.4	Semi-Authenticated Key Agreement Using Voice Records. . . . .	22
2.5	Man-In-The-Middle Attack (Simplified Version). . . . .	23
2.6	Distance-Bounding Protocol. . . . .	24
2.7	Key Agreement Protocol Using Distance Bounding. . . . .	25
3.1	Message Authentication Protocol Channels (General) Model . . . . .	27
3.2	Stronger Properties of Human Authenticated Channels . . . . .	30
3.3	SH Game. . . . .	33
3.4	FH Game. . . . .	33
3.5	SB Game. . . . .	34
3.6	FB Game. . . . .	34
3.7	Ideal Commitment: Commit Algorithm. . . . .	35
3.8	Ideal Commitment, Decommit Algorithm. . . . .	35
3.9	WCR game. . . . .	40
4.1	Generic One-Shot Attack . . . . .	42
5.1	Non-Interactive Channels Model . . . . .	45
5.2	Non-Interactive Message Authentication using a CRHF. . . . .	46
5.3	The MANA I Protocol. . . . .	47

5.4	Non-Interactive Message Authentication Based on a WCRHF. . . . .	48
5.5	Game Against the Proposed Protocol. . . . .	49
5.6	Reduced Game. . . . .	49
5.7	Reduction to the SB Game ( $\hat{c} = c$ ). . . . .	50
5.8	Reduction to the WCR Game ( $\hat{c} \neq c$ ). . . . .	50
6.1	SAS-based Message Authentication. . . . .	53
6.2	Connection to a Remote Host using a Conventional Protocol . . . . .	55
6.3	Connection to a Remote Host using the SAS-Based Protocol . . . . .	56
6.4	Implementation of the Vaudenay SAS-based Protocol . . . . .	58
6.5	Screen-shot of the Main Frame . . . . .	59
6.6	Screen-shot of the Frame on Alice Side (Verbose Mode) . . . . .	61
6.7	Screen-shot of the Frame on Bob Side (Verbose Mode) . . . . .	61
7.1	Three First Steps of the Recurrence. . . . .	81

# Appendix A

**Lemma 4.** *Let  $X$  and  $Y$  be two identically distributed independent random variables with distribution  $D$  over a support set  $S$ . We have*

$$\Pr[X = Y] \geq \frac{1}{\#S} \quad (7.1)$$

*with equality if and only if  $D$  is the uniform distribution.*

*Proof.* Let  $s$  be the size of the set  $S$ . We have

$$\begin{aligned} \Pr[X = Y] &= \sum_{S_i \in S} \Pr[X = S_i] \cdot \Pr[Y = S_i] \\ &= \sum_{S_i \in S} p_i^2 \end{aligned}$$

where  $p_i$  is  $\Pr[X = S_i]$ .

Let us write  $p_i = \frac{1}{s} + \rho_i$ . Thus, we obtain

$$\sum_{S_i \in S} p_i^2 = \sum_{S_i \in S} \left(\frac{1}{s}\right)^2 + 2 \sum_{S_i \in S} \frac{1}{s} \rho_i + \sum_{S_i \in S} \rho_i^2$$

Knowing that the sum of  $p_i$  equals to 1 we can easily deduce that the sum of  $\rho_i$  equals 0. Thus,

$$\sum_{S_i \in S} p_i^2 = \frac{1}{s} + \sum_{S_i \in S} \rho_i^2$$

The sum of  $\rho_i^2$  is greater or equal to 0. Note that it is equal to 0 if and only if all  $\rho_i$  are null, i.e.  $D$  is uniform.

Finally the probability searched is:

$$\Pr[X = Y] \geq \frac{1}{s}$$

with equality if and only if  $D$  is the uniform distribution. □



# Appendix B

**Lemma 5.** *We consider two sets of random values  $\{X_i\}$ , resp.  $\{Y_j\}$ , of size  $p$ , resp.  $q$ , where the elements are picked in a set  $S$  of size  $s$  with distribution  $D$ .*

*The probability of collision between the two sets is minimal when the distribution  $D$  is uniform.*

*Proof.* Let  $C_D^{p,q}$  the probability that there exists a  $X_i$  which corresponds to a  $Y_j$  given a distribution  $D$  along the set  $S$  of  $s$  elements, i.e. a collision occurred between the two sets.

$$C_D^{p,q} = \Pr [\{X_1, \dots, X_{Q_A}\} \cap \{Y_1, \dots, Y_{Q_B}\} \neq \emptyset]$$

Formally, we have to prove that

$$C_D^{p,q} \geq C_{U_s}^{p,q} \tag{7.2}$$

where  $U_N$  is the uniform distribution among a set of  $N$  possible elements.

We will proceed by first proving that Eq. 7.2 is true for a support  $S$  of one element. Then, we will prove by recurrence that Eq. 7.2 is true for any  $s$ , i.e. by proving that it is true for  $s$  elements assuming that it is true for  $s - 1$  elements.

**For a set  $S$  of one element**, i.e.  $s = 1$ , the result is straightforward since the only possible distribution is the uniform distribution and we have  $C_D^{p,q} = C_{U_1}^{p,q}$ .

**For a set  $S$  of any size  $n$** , we will prove that it is also true by assuming that Eq. 7.2 is true for  $n - 1$  or less elements. Let  $a$  an elements of  $S$  and  $t_a$  his probability, i.e.  $t_a = \Pr_D[X = a]$ . We define the distribution  $D'$  as

$$\Pr_{D'}[X = x] = \begin{cases} 0 & \text{if } x = a \\ \Pr_D[X = x | x \neq a] & \text{if } x \neq a \end{cases}$$

Considering the particular element  $a$ , we can write

$$\begin{aligned}
C_D^{p,q} &= \Pr [\{X_i\} \cap \{Y_j\} \neq \emptyset | a \notin \{X_i\}, a \notin \{Y_j\}] \\
&\quad \cdot \Pr[a \notin \{X_i\}, a \notin \{Y_j\}] \\
&+ \Pr [\{X_i\} \cap \{Y_j\} \neq \emptyset | a \in \{X_i\}, a \in \{Y_j\}] \\
&\quad \cdot \Pr[a \in \{X_i\}, a \in \{Y_j\}] \\
&+ \Pr [\{X_i\} \cap \{Y_j\} \neq \emptyset | a \in \{X_i\}, a \notin \{Y_j\}] \\
&\quad \cdot \Pr[a \in \{X_i\}, a \notin \{Y_j\}] \\
&+ \Pr [\{Y_i\} \cap \{Y_j\} \neq \emptyset | a \notin \{X_i\}, a \in \{Y_j\}] \\
&\quad \cdot \Pr[a \notin \{X_i\}, a \in \{Y_j\}]
\end{aligned}$$

which can be written as

$$\begin{aligned}
C_D^{p,q} &= C_{D'}^{p,q} \cdot (1-t_a)^p (1-t_a)^q \\
&+ 1 \cdot [1 - (1-t_a)^p][1 - (1-t_a)^q] \\
&+ \sum_{i=1}^p \left\{ C_{D'}^{p-i,q} \binom{p}{i} t_a^i (1-t_a)^{p-i} \right\} \\
&\quad \cdot [1 - (1-t_a)^p] (1-t_a)^q \\
&+ \sum_{i=1}^q \left\{ C_{D'}^{p,q-i} \binom{q}{i} t_a^i (1-t_a)^{q-i} \right\} \\
&\quad \cdot (1-t_a)^p [1 - (1-t_a)^q].
\end{aligned}$$

Noting that  $D'$  is distributed over  $n-1$  elements and assuming that Eq. 7.2 is true for  $n-1$  elements, we obtain

$$\begin{aligned}
C_D^{p,q} &\geq C_{U_{n-1}}^{p,q} \cdot (1-t_a)^p (1-t_a)^q \\
&+ 1 \cdot [1 - (1-t_a)^p][1 - (1-t_a)^q] \\
&+ \sum_{i=1}^p \left\{ C_{U_{n-1}}^{p-i,q} \binom{p}{i} t_a^i (1-t_a)^{p-i} \right\} \\
&\quad \cdot [1 - (1-t_a)^p] (1-t_a)^q \\
&+ \sum_{i=1}^q \left\{ C_{U_{n-1}}^{p,q-i} \binom{q}{i} t_a^i (1-t_a)^{q-i} \right\} \\
&\quad \cdot (1-t_a)^p [1 - (1-t_a)^q].
\end{aligned}$$

The right hand side of this inequality corresponds to the probability of collisions  $C_{D_0}^{p,q}$  where the distribution  $D_0$  is defined as

$$\Pr_{D_0}[X = x] = \begin{cases} t_a & \text{if } x = a \\ \frac{1}{n-1}(1-t_a) & \text{if } x \neq a \end{cases}.$$

Consequently, we obtain  $C_D^{p,q} \geq C_{D_0}^{p,q}$ . We repeat this step using the same reasoning but using another element. Let  $b \neq a$  an element of the set  $S$  and let  $t_b$  his probability over

$D_0$ , i.e.  $t_b = \Pr_{D_0}[X = b] = \frac{1}{n-1}(1 - t_a)$ . Proceeding as before, we obtain  $C_{D_0}^{p,q} \geq C_{D_1}^{p,q}$  where  $D_1$  is defined as

$$\Pr_{D_1}[X = x] = \begin{cases} t_b & \text{if } x = b \\ \frac{1}{n-1}(1 - t_b) & \text{if } x \neq b \end{cases} .$$

Finally, we obtain the following recurrence

$$C_D^{p,q} \geq C_{D_0}^{p,q} \geq C_{D_1}^{p,q} \geq \dots \geq C_{D_i}^{p,q} \geq \dots$$

where the distributions are defined as

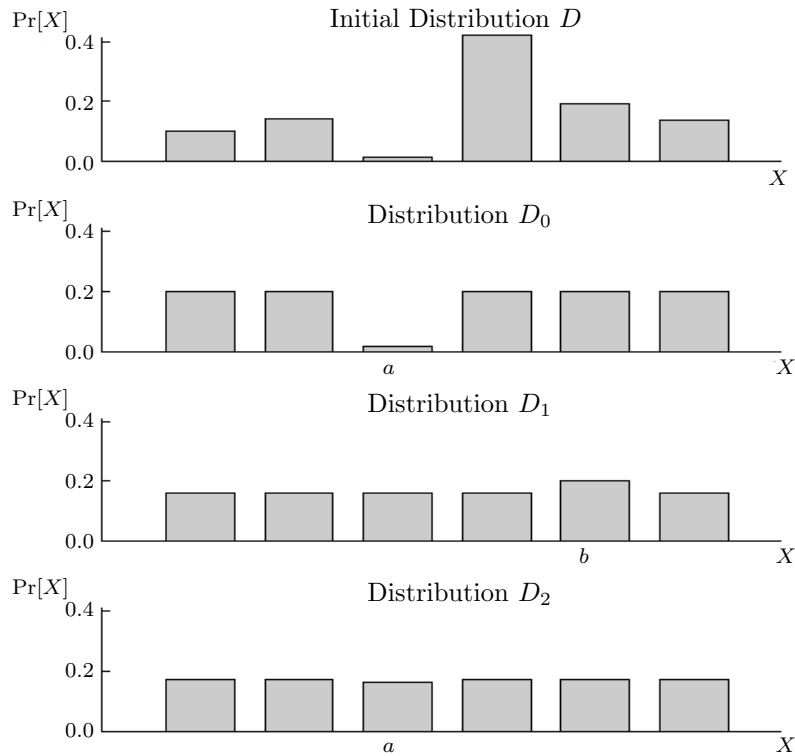
$$\Pr_{D_i}[X = x] = \begin{cases} t_i & \text{if } x = a_i \\ \frac{1}{n-1}(1 - t_i) & \text{if } x \neq a_i \end{cases}$$

with

$$a_i = \begin{cases} a & \text{if } i \text{ odd} \\ b & \text{if } i \text{ even} \end{cases}$$

and  $t_0$  is  $\Pr_D[X = a_i]$  and for  $i \geq 1$ ,  $t_i$  is  $\Pr_{D_{i-1}}[X = a_i]$ .

Fig. 7.1 represents on its top the original distribution  $D$ . The three first steps of the recurrence, i.e.  $D_0, D_1, D_2$ , are represented below it.



**Figure 7.1.** Three First Steps of the Recurrence.

It seems that the recurrence converges rapidly near the uniform distribution. Indeed, we have  $t_{i+1} = \frac{1}{n-1}(1-t_i)$  and thus  $t_i = \frac{1}{n} + (-\frac{1}{n-1})^i \cdot (t_0 - \frac{1}{n})$  which converges to  $\frac{1}{n}$  when  $n > 1$ . We can finally write

$$C_D^{p,q} \geq \dots \geq C_{D_t}^{p,q} \xrightarrow{t \rightarrow \infty} C_{U_n}^{p,q}.$$

We conclude that any probability of collision  $C_D^{p,q}$ , where  $D$  is any distribution along  $S$ , is bigger or equal to the probability of collision  $C_{U_s}^{p,q}$  which uses uniformly distributed SAS.  $\square$

In conclusion, any authentication protocol must use uniformly distributed SAS. Any other distribution allows adversaries to find collisions with higher probability or smaller complexity and thus the protocol is less secure.