Secure Session Framework: An Identity-based Cryptographic Key Agreement and Signature Protocol

D i s s e r t a t i o n

zur Erlangung des
Doktorgrades der Naturwissenschaften
(Dr. rer. nat.)

dem Fachbereich Mathematik und Informatik
der Philipps-Universität Marburg
vorgelegt von

Christian Schridde
geboren in Peine

Marburg, 2010

Vom Fachbereich Mathematik und Informatik der
Philipps-Universität Marburg als Dissertation am

07.05.2010

angenommen.

1. Gutachter: Prof. Dr. Bernd Freisleben, Philipps-Universität Marburg
2. Gutachter: Prof. Dr. Matthew Smith, Leibniz Universität Hannover

Datum der mündlichen Prüfung: 05.07.2010

Erklärung

Ich versichere, dass ich meine Dissertation

Secure Session Framework: An Identity-based Cryptographic Key Agreement and Signature Protocol

selbständig, ohne unerlaubte Hilfe angefertigt und mich dabei keiner anderen als der von mir ausdrücklich bezeichneten Quellen und Hilfen bedient habe. Die Dissertation wurde in der jetzigen oder einer ähnlichen Form noch bei keiner anderen Hochschule eingereicht und hat noch keinen sonstigen Prüfungszwecken gedient.

------------------------	-----------------------
Ort, Datum	Unterschrift

Acknowledgments

I would like to acknowledge the help of several people during the course of this doctoral thesis.

First of all, I would like to thank my thesis supervisor Prof. Dr. Bernd Freisleben for the scope for development he gave me over the years and his permanent support in all issues relevant for conducting this research.

I would also like to thank my former colleague Prof. Dr. Matthew Smith from whom I learned a lot during all the discussions we had while we shared a room during four years of work.

Also, special thanks go to my colleague Dr. Ralph Ewerth for all the interesting and funny conversations during the breaks in the corridor.

My thanks also go to the rest of the people in the Distributed Systems Group in Marburg: Dr. Markus Mathes, Thilo Stadelmann, Tim and Kay Dörnemann, Dominik Seiler, Matthias Schmidt, Nils Fallenbeck, Ernst Juhnke, Roland Schwarzkopf, Markus Mühling, and last but not least, Mechthild Kessler.

Finally, I would like to thank my parents Gerhard and Ursula Schridde, my sister Tania and my girlfriend Nicole for being part of my life.

Zusammenfassung

Die vorliegende Dissertation beschäftigt sich mit der Methode der *identitätsbasierten Verschlüsselung*. Hierbei wird der Name oder die Identität eines Zielobjekts zum Verschlüsseln der Daten verwendet. Diese Eigenschaft macht diese Methode zu einem passenden Werkzeug für die moderne elektronische Kommunikation, da die dort verwendeten Identitäten oder Endpunktadressen weltweit eindeutig sein müssen. Das in der Arbeit entwickelte identitätsbasierte Schlüsseleinigungsprotokoll bietet Vorteile gegenüber existierenden Verfahren und eröffnet neue Möglichkeiten. Eines der Hauptmerkmale ist die komplette Unabhängigkeit der Schlüsselgeneratoren. Diese Unabhängigkeit ermöglicht es, dass verschiedene Sicherheitsdomänen ihr eigenes System aufsetzen können. Sie sind nicht mehr gezwungen, sich untereinander abzusprechen oder Geheimnisse auszutauschen. Auf Grund der Eigenschaften des Protokolls sind die Systeme trotzdem untereinander kompatibel. Dies bedeutet, dass Anwender einer Sicherheitsdomäne ohne weiteren Aufwand verschlüsselt mit Anwendern einer anderen Sicherheitsdomäne kommunizieren können. Die Unabhängigkeit wurde ebenfalls auf ein Signatur-Protokoll übertragen. Es ermöglicht, dass Benutzer verschiedener Sicherheitsdomänen ein Objekt signieren können, wobei auch der Vorgang des Signierens unabhängig sein kann.

Neben dem Protokoll wurde in der Arbeit auch die Analyse von bestehenden Systemen durchgeführt. Es wurden Angriffe auf etablierte Protokolle und Vermutungen gefunden, die aufzeigen, ob oder in welchen Situationen diese nicht verwendet werden sollten. Dabei wurde zum einen eine komplett neue Herangehensweise gefunden, die auf der (Un-)Definiertheit von bestimmten Objekten in diskreten Räumen basiert. Zum anderen wurde die bekannte Analysemethode der Gitterreduktion benutzt und erfolgreich auf neue Bereiche übertragen.

Schlussendlich werden in der Arbeit Anwendungsszenarien für das Protokoll vorgestellt, in denen dessen Vorteile besonders relevant sind. Das erste Szenario bezieht sich auf Telefonie, wobei die Telefonnummer einer Zielperson als Schlüssel verwendet. Sowohl GSM-Telefonie als auch VoIP-Telefonie werden in der Arbeit untersucht. Dafür wurden Implementierungen auf einem aktuellen Mobiltelefon durchgeführt und bestehende VoIP-Software erweitert. Das zweite Anwendungsbeispiel sind IP-Netzwerke. Auch die Benutzung der IP-Adresse eines Rechners als Schlüssel ist ein gutes Beispiel, jedoch treten hier mehr Schwierigkeiten auf als bei der Telefonie. Es gibt beispielsweise dynamische IP-Adressen oder die Methode der *Network Address Translation*, bei der die IP-Adresse ersetzt wird. Diese und weitere Probleme wurden identifiziert und jeweils Lösungen erarbeitet.

Abstract

Cryptographic protocols are used to encrypt data during their transmission over a network or to store it on a data carrier. This thesis is about the method of identity-based encryption. In this form of encryption, the name or identity of the target subject is used to encrypt the data. This property makes it a perfect tool for modern electronic communication, because all involved identities and endpoint addresses (e.g. IP addresses) have to be unique worldwide and must be known in order to establish a communication. The identity-based key agreement protocol that has been invented in this thesis has several advantages compared to existing schemes. One important property is its complete independence of key generators. This independence allows each participating security domain to set up and maintain its own key generator. They are not forced to agree on a common setup or a common secret anymore. Due to the properties of the protocol, the security domains are still compatible to each other. Users from one security domain can communicate with users from another security domain using encryption. This new property of independence is also carried over to a signature protocol. It allows users from different security domains to sign a certain object. Additionally, the act of signing is independent and the signers do not need to communicate with each other.

Apart from the protocol and its security proofs with respect to standard definitions from the literature, the thesis contains an analysis of existing schemes. Attacks on known protocols and assumptions are presented, and it is shown under which circumstances these become insecure. On the one hand, a completely new approach that is based on defined or rather undefined objects in discrete structures is used. On the other hand, the method of lattice based reduction is successfully applied to the new area of secret sharing schemes.

Finally, application scenarios for the protocol are presented. These scenarios are chosen such that the advantages of the protocol become apparent. The first application is telephony, GSM as well as Voice over IP (VoIP). In this case, the telephone number of the callee is used as the encryption key. Implementations on a modern mobile phone as well as within existing Voice over IP software are presented. The second application is IP networks. Here, the IP address of a communication unit is used as the encryption key. However, in this case, there are more problems than in the GSM/VoIP case, e.g., dynamic IP addresses or network address translation (NAT) where an IP address is substituted by another one. These are only two problems out of several for which solutions are presented.

Contents

List of Figures

List of Algorithms

1 Introduction

"Why should you care if you have nothing to hide?"
J. Edgar Hoover

Already at the time when nobody has even thought about electronic communication, hiding or altering data was a usual method to prevent sensitive information from falling into foreign hands. Short messages on the bottom of a filled water bucket, scratched words on a scalp or a simple character substituting scheme were methods that have already been used before Christ. In both cases, when receiving a message that is hidden *in* something (stenographic) or encrypted *with* something (cryptographic), the sender must have some previous knowledge. He must either know that there is a message hidden, in order to reveal it, or he must know the information how to decrypt the entire data. Receiving unexpected messages from unknown senders confronts a sender with a problem, since he does not know if there is something between the lines or how to reconstruct the message. Consequently, the sender and the receiver must have communicated once before in order to generate a common knowledge. Therefore, they either have met each other personally or they used another already existing secure communication channel. In the early periods of time, long travels for personal meetings were the usual way to exchange information. In the age of electronic communication, this investment nullifies the basic idea of the new and fast communication form. Despite the advancements in cryptologic research, the core problem of encryption remains, that is, all participants must have a key (the same) and they must receive it protected from foreign access. Retrospectively, this kind of encryption is called *Symmetric Cryptography*. It earns its name from the fact that every participant who has access to the encrypted channel possesses the *same* key. This means, encrypting and decrypting is done using the same secret.

1.1 Public Key Cryptography

A conceptually new approach has been proposed by Diffie and Hellman in the seventies of the 20-th century [40]. They have introduced a concept called *Public Key Cryptography* (Abbr: PKC). The important improvement was that they did decryption and encryption with different keys instead of the same. One key is a private key that is used to decrypt incoming messages and has to be stored in a secure way. The other key is a public key that is used by remote parties to encrypt messages addressed to the key's owner. The usage of two keys forms the name *Asymmetric Cryptography*, since the symmetry in encryption and decryption has been given up. The advantage is that the key to encrypt a message can be distributed publicly, since it can not be used to decrypt a message. Using PKC, participants do not have to communicate in order to negotiate a key, but a user can publish his encryption key, to be accessible for all potential communication partners.

To make this principle work, their key idea was to use a mathematical object, called *one-way function*. A one-way function f has the property to be easy to compute but difficult to invert. This means $f(x) = y$ can be computed easily but $f^{-1}(y) = x$ is hard to find, i.e. in non-polynomial time. As an instance for the one-way function, Diffie and Hellman used *discrete exponentiation*. This is the task to compute the integer r in the congruence $f_g(e) \equiv g^e \equiv r \pmod{p}$, given g, e, p, and can be done in parts of a second on modern computers. The computation of $f_g^{-1}(r) = e$ given (g, r, p) is infeasible in a reasonable amount of time and is called the *Discrete Logarithm Problem*. Note that the owner of the one-way function instance knows the integer e, thus he knows the inversion of the function per default. The fact that there exists no known algorithm that can invert this function in a fast way (in polynomial time) allows to publish the one-way function (and thus the public key) without any weakening consequences regarding the protocol. As long as the *Discrete Logarithm Problem* stays a problem, the protocol is safe, since also a potential adversary can not overcome this burden. The protocol has become famous as the Diffie-Hellman protocol and can be classified as a *Key Agreement Protocol*. The aim of a key agreement protocol is to let participants agree onto a common key. Since it has not been designed to encrypt messages, it is not practical and sufficient in all situations.

A short time later, Rivest, Shamir and Adleman introduced the first public key encryption system [102], called RSA. They adopted the Diffie-Hellman approach, but they utilized another one-way function. Instead of discrete exponentiation, they used *multiplication*. To make multiplication a one-way function, the factors have to be chosen in a special way. They proposed to use two large primes P and Q to build the integer $N = PQ$. The inverse operation, called *factorization*, is an infeasible operation for factors of sufficient size. In the RSA encryption scheme, the public key of a participant is the tuple $(N = PQ, e)$, whereof e is chosen randomly, but with the constraint that it neither divides $P-1$ nor $Q-1$, which is equivalent to the statement $\gcd(e, \varphi(N)) = 1$. The private key d is then uniquely determined by the congruence $ed \equiv 1 \pmod{\varphi(N)}$. At this point, the one-wayness of multiplication is utilized, since no one can compute the private key d from public integers e and N without knowing $\varphi(N)$. However, if someone is able to compute $\varphi(N)$ or d, he is also able to factorize N, thus he finds an inversion of the one-way function. For the actual message encryption, the authors used another one-way function that is associated with the first one-way function but is often formulated independently. Rivest, Shamir and Adleman have utilized that it is infeasible to compute roots in $\mathbb{Z}_N$ if the the factorization of N can not be found. They transformed a message m into an integer M out of $\mathbb{Z}_N$ and executed the one-way function $f(M, e) \equiv M^e \equiv C \pmod{N}$, which makes C the ciphertext of M. Since computing roots in $\mathbb{Z}_N$ is assumed to be NP-hard (the integer M is the e-th root of the ciphertext C), the ciphertext is protected. To decrypt a message, the ciphertext is raised to the power of the secret key, which yields: $C^d \equiv M^{ed} \equiv M^{1+\varphi(N)k} \equiv M \pmod{N}$.

PKC is one of the most important inventions in cryptography. However, the problem of key distribution has not been solved completely. An adversary could still use the phase of public key distribution to by-pass the encryption scheme. The reason is that PKC provides no binding between the public key and its owner. A *Man-In-The-Middle Attack* (Abbr: MITMA) can utilize this missing binding to subvert public key communication *without* breaking the underlying one-way function. A simple example helps to understand the problem: Suppose in a public key scenario an adversary $\mathcal{A}$ manages to substitute Alice's public key with his own public key. This can be easier than its sounds. Public keys are often stored on public ring servers, where anyone can publish a key under an arbitrary name. If $\mathcal{A}$ publishes a key marking Alice as its owner, Bob can accidentally grab this wrong key, believing to have Alice's original key. Furthermore, assume $\mathcal{A}$ has access to Alice's mailbox (for administrators on certain systems this is not unusual) or he has access to the network connection between Alice and Bob. If now user Bob uses the fake public key of Alice and encrypts a message, $\mathcal{A}$ can decrypt the message in Alice's mailbox, since he is the correct owner of the public key and thus $\mathcal{A}$ has also the corresponding private key. After copying the plaintext, he re-encrypts the message using Alice's original public key and leaves the message in Alice's inbox. Alice will not notice anything, since she only comes upon a correctly encrypted message in her mailbox.

1.2 Identity-based Cryptography

In 1984, Shamir, one of the inventors of RSA, published a paper [116] in which he proposed an idea to overcome the problem of MITMA. He called his approach *Identity-based Cryptography* (Abbr: IBC). His idea is to use the identity of a participant itself as the public key rather than a specially crafted integer. This identity can be each unique identifier of a participant that must be publicly known or can be retrieved in a reliable manner. Here, the sender has to know something, namely the identity of the receiver. However, it is much easier to get this identifier or to verify this identifier by a simple look, rather than to be confronted with large columns of digits that have no relationship to its owner. An often used example are e-mail addresses. If a user wants to send an encrypted message, he at least has to know the e-mail address of its receiver. Otherwise, he obviously can not even send the message. Thus, if the sender knowns the receiver's e-mail address, he knows his public key as well. Another example is to use telephone numbers. If a participant is not sure about the number and the voice that picks up on the other side of the line does not sound like the voice expected, something is wrong with the phone number, thus something is wrong with the encryption key. However, if the person responding is the person intended to be called, the correct number has been dialed and thus a secure channel has been established. In this way, the user directly gets feedback about the status of the encryption.

If an arbitrary string is used as a public key, then there also must be a corresponding private key for each string. Obviously, a user can not generate his private key on his own. If this was possible, than either also an adversary can generate the private key, or the user has to utilize some additional one-way function. In the latter case, the one-way function must be added to the user's public key (the identity) and contradicts the concept of a publicly known identifier. Thus, in order to generate the private keys, a trusted generator is used, similar to the well adopted certificate authority in public key infrastructures. These generators are called Identity Private Key Generators (ID-PKG). Each trusted generator possesses a set of public parameters that are shared among all participants. They can, e.g., be hardcoded into the protocol's implementation. These shared, public parameters define the basic parameters of the protocol, like the specification of the used modulus, involved fixed exponents or utilized hash functions. These public parameters are based on one-way functions, where the inversion is only known to the trusted generator, which enables the generator to generate the private keys for each participant.

Since Boneh and Franklin [22] have published the first provably secure identity-based encryption scheme in 2002, identity-based encryption has been evolved into an important part of cryptography in the last years. Various encryption and key agreement scheme have been developed. However, there are several open issues that have not been solved satisfactorily.

1.3 Contributions

The contributions of this thesis are as follows:

- Based on well known assumptions in cryptography, a novel identity-based key agreement scheme, called *Secure Session Framework* (Abbr: SSF), is proposed.

- SSF is the first identity-based key agreement scheme that can handle completely independent ID-PKGs. In the existing literature, ID-PKGs are either forced to agree on a common secret or to form some kind of hierarchical structure.

- Based on the key agreement scheme, an identity-based multi-signature scheme is proposed that also supports independent ID-PKGs as well as non-interactive signatures.

- Several attacks relevant for the presented scheme are analyzed. It is shown that the Φ-Hiding assumption can be broken with non-negligible probability in a specific environment. What makes this result surprising is that the Φ-Hiding assumption is deeply associated with the factorization problem that is still one of the major problems in cryptology.

- Secret sharing schemes are analyzed. It is shown that if a secret sharing scheme that is based on the chinese remainder theorem is used to split the integer $\varphi(N)$ in several pieces, the secret sharing scheme can be broken under certain circumstances. Using this result, the protocol by Iftene and Grindei [65] can be proven to be insecure if enough malicious users collaborate.

- Finally, two application scenarios for the proposed SSF scheme are presented. The first scenario is the application to IP networks and the second the application the GSM and VoIP communication. In this context, the problems NAT traversal, dynamic IP addresses, and the distribution of involved keys/parameters are solved.

1.4 Publications

The following publications have been produced during the work on this thesis:

- Christian Schridde and Matthew Smith and Tim Dörnemann and Ernst Juhnke and Bernd Freisleben, An Identity-Based Security Infrastructure for Cloud Environments, *2010 IEEE International Conference on Wireless Communications, Networking and Information Security* (2010, Peking, China), (accepted for publication), IEEE Press

- Christian Schridde and Matthew Smith and Björn Agel and Bernd Freisleben, Secure Mobile Communication via Identity-based Cryptography and Server-aided Computations, *The Journal of Supercomputing*, (accepted for publication), Spring-Verlag, 2010

- Christian Schridde and Matthew Smith and Bernd Freisleben, Non-Interactive Multi-Signatures with Multiple Independent Identity Key Generators, 2009 (*submitted for publication*)

- Christian Schridde and Matthew Smith and Bernd Freisleben, Partial Key Exposure Attacks on Secret Sharing Schemes, 2009 (*submitted for publication*)

- Christian Schridde and Matthew Smith and Bernd Freisleben, TrueIP: Prevention of IP Spoofing Attacks using Identity-based Cryptography, *SIN'09 - Proceedings of the 2nd International Conference on Security of Information and Networks* (2009, Gazimagusa, North Cyprus), pp.128-137, ACM Press

- Matthew Smith and Christian Schridde and Bernd Freisleben, Securing Mobile Phone Calls with Identity-Based Cryptography, *ISA'09 - Proceedings of the 3rd International Conference on Information Security and Assurance* (2009, Seoul, Korea), vol. 5576 of *Lecture Notes in Computer Science*, pp. 124-134, Springer

- Matthew Smith and Christian Schridde and Bernd Freisleben, Identity-Based Cryptography for Securing Mobile Phone Calls, *HWISE - Proceedings of the 5th IEEE International Workshop on Heterogeneous Wireless Sensor Networks* (2009, Bradford, UK), pp. 29-33, IEEE Press

- Christian Schridde and Bernd Freisleben, On the Validity of the Phi-Hiding Assumption in Asymmetric Cryptographic Protocols, *ASIACRYPT - Advances in Cryptology* (2008, Melbourne, Australia), vol. 5350 of *Lecture Notes in Computer Science*, pp. 344-354, Springer

- Christian Schridde and Matthew Smith and Bernd Freisleben, An Identity-Based Key Agreement Protocol for the Network Layer, *SCN'08 - Proceedings of the 6th International Conference on Security and Cryptography for Networks* (2008, Amalfi, Italy), vol. 5229 of *Lecture Notes in Computer Science*, pp. 409-422, Springer

- Matthew Smith and Christian Schridde and Bernd Freisleben, Securing Stateful Grid Servers through Virtual Server Rotation, *HPDC'08 - Proceedings of the 17th ACM/IEEE International Symposium on High Performance Distributed Computing* (2008, Boston, MA), pp. 11-23, ACM Press

- Matthew Smith and Matthias Schmidt and Nils Fallenbeck and Tim Dörnemann and Christian Schridde and Bernd Freisleben, Secure On-Demand Grid Computing, *Journal of Future Generation Computer Systems*, pp. 315-325, Elsevier, 2008

- Matthew Smith and Matthias Schmidt and Nils Fallenbeck and Christian Schridde and Bernd Freisleben, Optimising Security Configurations with Service Level Agreements, *Proceedings of the 7th International Conference on Optimization: Techniques and Applications* (2007, Kobe, Japan), pp. 367-381

- Christian Schridde and Hans-Joachim Picht and Michael Heidt and Matthew Smith and Bernd Freisleben, Secure Integration of Desktop Grids and Compute Clusters Based on Virtualization and Meta-Scheduling, *Proceedings of the German e-Science Conference* (2007, Baden-Baden, Germany), pp. 23-28

- Thomas Barth and Kay Dörnemann and Tim Dörnemann and Bernd Freisleben and Thomas Friese and Manfred Grauer and Jürgen Jakumeit and Julian Reichwald and Christian Schridde and Matthew Smith and Frank Thilo, Supporting Engineering Processes Utilizing Service-Oriented Grid Technology, *Proceedings of German e-Science Conference* (2007, Baden-Baden, Germany), pp. 1-10

- Patent Applications:

  1. Nr. 10 2007 033 846.7,
     Applicant: Christian Schridde, Dr. Matthew Smith, Prof. Dr. Bernd Freisleben, Ansgar Kewitz
     Title: Verschlüsselungssystem

  2. Nr. 10 2007 033 848.3,
     Applicant Christian Schridde, Dr. Matthew Smith, Prof. Dr. Bernd Freisleben, Ansgar Kewitz
     Title: Verschlüsselungssystem Spoofing und VPN

  3. Nr. 10 2007 033 845.9,
     Applicant Christian Schridde, Dr. Matthew Smith, Prof. Dr. Bernd Freisleben, Ansgar Kewitz
     Title: Verschlüsselungssystem VoIP

  4. Nr. 10 2007 033 847.5,
     Applicant Christian Schridde, Dr. Matthew Smith, Prof. Dr. Bernd Freisleben, Ansgar Kewitz
     Title: Verschlüsselungssystem NAT und DHCP

1.5 Organization

The thesis is organized as follows:

In Chapter 2, the fundamental problems that are referred to later in the thesis are defined. Then, standard definitions about security attributes regarding encryption and signature systems as well as a definition for a secure and authenticated key agreement protocol are given. Finally, the Random Oracle Model is introduced.

In Chapter 3, the new IBC scheme is presented. It begins with a short introduction to related work and open problems regarding identity-based cryptography. Next, the four main algorithms for the basic key agreement case and two additional algorithms for the extension to the multi-authority case are presented. Finally, the algorithms that are used to build and verify a signature are presented, both in the basic case and in the multi-authority, multi-signature case.

In Chapter 4, security proofs of the presented algorithms are presented. It is shown that the presented key agreement algorithm (single and multi authority) is secure in the Canetti-Krawzky Model (Abbr: CKM) [30, 31], which makes SSF a secure and authenticated identity-based key agreement protocol. Furthermore, a proof for the signature algorithm in the random oracle model is provided and it is shown that its security can be reduced to a well known NP-hard problem.

Chapter 5 is about related attacks. In particular, it is shown that the Φ-Hiding assumption can be broken with non-negligible probability when it is applied to *multi-power* moduli. This is of also of general interest in the area of cryptography. Next, it is demonstrated that when a threshold-based approach is used to create the private keys in the way that the master secret key is shared across several key generators, the corresponding secret sharing scheme has to be chosen carefully. Otherwise, the secret key can be recovered in polynomial time using lattice based reduction methods.

In Chapter 6, it is shown how the presented algorithm applies to real world scenarios. Therefore, problems that occur in IPv4 networks, SIP environments and GSM communication are presented. Problems regarding private key distribution, public shared parameter distribution, key revocation, dynamic identities and identity translation are discussed. The end of this chapter is about an optimization that uses server-aided cryptography to speed up expensive computations on less powerful devices, e.g., smartphones or PDAs.

In Chapter 7, measurements of the proposed scheme are presented. The measurements concern the presented algorithms and are given for different devices. Finally, the efficiency of the scheme is compared with another scheme from the literature.

In Chapter 8, the thesis is concluded with a summary and future work.

2 Fundamentals of Cryptography

"Equations, however impressive and complex cannot arrive at the truth if the initial assumptions are incorrect."
Arthur C. Clarke

2.1 Introduction

Public key cryptography is feasible, because there are still mathematical problems that can not be solved in an efficient way. PKC uses this fact and faces potential adversaries with these problems. If an attacker finds a practical way to solve the underlying mathematical problem, he will be able to break the protocol. However, the reverse direction is not true for most of the existing protocols. This means, that it is not known if a protocol can only be broken if the NP-hard problem it is based on can be solved or if the protocol can be broken in a completely other way.

Definition 2.1.1 (negligible function) *A function* $\mu(x) : \mathbb{N} \to \mathbb{R}$ *is said to be negligible if for any* $c > 0$ *there is an* N_c*, such that for all* $x > N_c$*,* $|\mu(x)| < \frac{1}{x^c}$ *holds.*

Negligible functions are often used in conjunction with the success probability of an attacker to break a cryptographic system. Therefore, an *advantage function* describes the success probability of an attacker in the long run to break the entire system. Regarding a classical assumption, this concerns, e.g., to get the private key from the public key. When applied to a decisional assumption, the advantage function compares the success probability versus random guessing.

When using a mathematical problem, it can be abstracted as a function f that must be inverted in order to break the system. In general, a function that is infeasible to invert is called a *one-way function.* A hash function is a typical example for a (compress) one-way function. In PKC special one-way functions are used. These special functions possess a trapdoor, which is hard to find, usually in non-polynomial time, whereby they earned the name *one-way functions with trapdoor.* However, if the trapdoor is known, the function can be inverted easily. The knowledge of this trapdoor is used as a user's secret key.

Definition 2.1.2 (One-Way Function) *A function* $\mathsf{f} : \{0,1\}^+ \to \{0,1\}^+$ *is a one-way function if*

- *efficient evaluation: there exists a polynomial-time algorithm* Eval *such that* $\mathsf{Eval}(x) = \mathsf{f}(x)$ *for all* $x \in \{0,1\}^+$
- *one-wayness: for any probabilistic polynomial-time algorithm* $\mathcal{A}$*, the inversion probability*
$$\mathsf{Inv}^{\mathsf{f}}_{\mathcal{A}}\ (n) = \Pr[\mathcal{A}\ (1^n, \mathsf{f}(x)) \in \mathsf{f}^{-1}(\mathsf{f}(x))]$$
is negligible in n*, where the probability is taken over* $x \in_R \{0,1\}^n$*. If additionally* $\mathsf{f}(\{0,1\}^n) = \{0,1\}^n$ *for all* $n \in \mathbb{N}$*,* f *is a one-way permutation.*

Unfortunately, it is not known if the functions that are presented next are indeed one-way functions. It is also not known if one-way functions exists at all. However, until now, no algorithms are known that can invert the functions in polynomial time in the standard model. Thus, the correct name would be to say *candidates for one-way functions with trapdoor*. But since it is widely believed that these functions are one-way functions, we overtake the nomenclature from the existing literature and also use only the name one-way function during the rest of the thesis. It should be remarked that in the quantum model, Shor's Algorithm [118] solves some of the problems in polynomial time.

Weak and Strong Assumptions. A computational assumption is often said to be weaker or stronger than another assumption. In general, a system should be based on the weakest possible assumption.

An assumption A *is weaker than an assumption* B *if the security of* B *depends on the security of* A*. Formally, this is written as* $\mathsf{A} \Rightarrow \mathsf{B}$*.*

Before reaching the actual problems, a few useful notations are introduced. Let $\mathbb{Z}_N := \{0, 1, 2, \ldots, N-1\}$ denote the ring of integer modulo N whereof N is a positive integer. The set $\mathbb{Z}_N^* \subset \mathbb{Z}_N$ consists of all integers that are coprime to N, i.e., $\mathbb{Z}_N^* = \{x \in$

$\mathbb{Z}_N | \gcd(x, N) = 1\}$. The number of elements in $\mathbb{Z}_N^*$ is $\varphi(N)$, with $\varphi(N)$ being Euler's Totient function, defined as:

$$N = \prod p_j^{\nu_j} \Rightarrow \varphi(N) = \prod p_j^{\nu_j - 1}(p_j - 1)$$

Another symbol that is utilized, is the *Jacobi symbol*, which can be evaluated efficiently even for composite numbers with unknown factorization [42]. The Jacobi symbol $J_p(r)$, for P prime, generalizes the Legendre symbol and states information about quadratic residues: If $a^2 \equiv r \pmod p$, for given integers r and p, has a solution in a, then $J_p(r) = 1$, otherwise $J_p(r) = -1$ (if $\gcd(p, r) > 1$, then $J_p(r) = 0$). For composite odd integers, the Jacobi symbol is defined as $J_N(r) = \prod_{j=1}^{m} J_{p_j}(r)^{\nu_j}$, if $N = p_1^{\nu_1} \dots p_m^{\nu_m}$.

Last, the Chinese Remainder Theorem (CRT) is recalled, that is utilized several times in the thesis.

Theorem 2.1.3 (Chinese Remainder Theorem) *Let* $n_1, \dots, n_k$ *be integers with* $\gcd(n_i, n_j) = 1$ *whenever* $i \neq j$. $N = \prod_{i=1}^{k} n_i$ *and* $a_1, \dots, a_k$ *be integers. Then, the system*

$$\begin{aligned} x &\equiv a_1 \pmod{n_1} \\ x &\equiv a_2 \pmod{n_2} \\ \dots \\ x &\equiv a_k \pmod{n_k} \end{aligned}$$

has exactly one solution $x \in \mathbb{Z}_N$.

The CRT is a very helpful theorem in cryptography, since it allows to characterize an integer by the help of a set of smaller integers. Since big integers are just the key to success of cryptography, it is very grateful to have a tool at hand that can degrade their size.

2.2 NP-hard Problems

Definition 2.2.1 (Integer Factorization Problem (Abbr: IFP)) *Given an integer* $N = p \cdot q$, *where* p, q *are primes. Find* p *and* q *in probabilistic polynomial time.*

The most established cryptographic hardness assumption is without doubt the IFP. The IFP as presented above is a special form, where the integer N only has two unknown primes and it is the usual way it is used in cryptography. It is obvious that factorization is not always a hard problem. Small integers as well as integers with prime factors of a special form or limited size can be factored in a short or reasonable time. This leads to the fact that primes used in a cryptosystem (e.g., RSA) should not be selected randomly, but in such a way, that they resist all known attacks and factorization approaches. Today, the two most efficient algorithms to factor generic integers known are the general number field sieve [74] and the quadratic sieve [100].

If N is of the described form ($N = pq$), the value of Euler's Totient function becomes $\varphi(N) = \varphi(pq) = (p-1)(q-1)$. Note, that if this value (and N) is known to some adversary, he obtains the system

$$\begin{aligned} p \cdot q &= N \\ (p-1)(q-1) &= \varphi(N) \end{aligned} \quad \Rightarrow \quad p^2 + (\varphi(N) - N - 1)p + N = 0$$

which can be solved in p instantly. Thus, it is crucial for a system which relay on the IFP to keep the value $\varphi(N)$ as well as the primes safe.

Definition 2.2.2 (RSA Assumption (Abbr: RSAA)) *Given the triple* (N, e, C) *with* N *being an integer with unknown factorization,* $\gcd(e, \varphi(N)) = 1$*,* $2 < e < N$ *and* $0 < C < N$*. For any probabilistic polynomial time adversary* $\mathcal{A}$*, the probability to find the integer* M *that satisfies* $M^e \equiv C \pmod N$ *is negligible.*

The RSAA can be directly derived from the definition of the RSA encryption system, where C is the ciphertext and the tuple (N, e) is the involved public key. The RSAA states that it is impossible to get the plaintext from an RSA ciphertext when only the public key and the ciphertext is known. Since the RSAA can be broken if N can be factored, the RSAA is a stronger assumption than the IFP, thus $\mathsf{IFP} \Rightarrow \mathsf{RSAA}$. It is still not known whether the security of the RSAA does rely on the IFP only. There are publications that show arguments in both direction, e.g., [23], [66] or [3].

An attacker who had success in computing $\varphi(N) = (p-1)(q-1)$ can compute the inverse element of e in $\mathbb{Z}^*_{\varphi(N)}$ (the decryption key d) which can be written as

$$ed = 1 + \varphi(N)k.$$

The Extended Euclidean Algorithm is a sufficient tool for this purpose (see [125], Section 3.2). A theorem of Euler (see for instance [70] Proposition I.3.5) states that for all $m \in \mathbb{Z}_N^*$, $m^{\varphi(N)} \equiv 1 \pmod N$ holds. Thus, the final computation of $C^d \equiv M^{ed} \equiv M^{1+\varphi(N)k} \equiv M \pmod N$ reveals M and breaks the RSAA.

Note that the definition can be also done using the term of negligible function and advantage function. However, these kind of definition will only be used when it is talked about decisional problems.

Definition 2.2.3 (Strong RSA Assumption (Abbr: sRSAA)) *Given the tuple* (N, C) *with* N *being an integer with unknown factorization and* $0 < C < N$*. For any probabilistic polynomial time adversary* $\mathcal{A}$*, the probability to find the integer tuple* (M, e) *that satisfies* $M^e \equiv C \pmod N$ *is negligible.*

The sRSAA is often used to proof the security of signature schemes and is, as the name suggests, a stronger assumption than the RSAA (IFP $\Rightarrow$ RSAA $\Rightarrow$ sRSAA). It states that it is not only infeasible to compute the e-th root (with e fixed) in $\mathbb{Z}_N$, but it is even infeasible to compute any root of a given integer when the factorization of N is unknown. Thus, if an adversary is able to compute the e-th root (e fixed) of a random integer M, he can also break the sRSAA, however the converse is not true.

Definition 2.2.4 (Φ-Hiding Assumption (Abbr: PHA)) *Let* $p_1 > 2$ *and* $p_2 > 2$ *be two random, small primes and* N *be an integer that is constructed such that exactly one of these two primes divides* $\varphi(N)$*. Then, for any probabilistic polynomial time adversary* $\mathcal{A}$ *, the advantage function, if* p_b *divides* $\varphi(N)$*,* $b \in \{1, 2\}$

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{PHA}}(k) := \left| \Pr[\mathcal{A}(N, p_1, p_2) = b] - \frac{1}{2} \right|$$

is a negligible function in k*.*

The PHA is a relatively new assumption and is stronger than the IFP, thus IFP $\Rightarrow$ PHA. The IFP says that it is impossible to compute the integer $\varphi(N)$ in reasonable time if N is chosen properly. The PHA states that it is also impossible to reveal any properties about the prime factors of $\varphi(N)$. The PHA was proposed by Cachin et al. [27] in the context of an algorithm for private information retrieval, based on this assumption. In Chapter 5, it will be shown that the PHA is not valid for all integers N with unknown factorization.

Definition 2.2.5 (Discrete Logarithm Problem (Abbr: DLP)) *Let* $\mathbb{G}$ *be a finite, cyclic subgroup of* $\mathbb{Z}_P^*$ *with a generator* g *and a large prime number* P*. For any proba-*

bilistic polynomial time adversary $\mathcal{A}$ who knows the triple (g, r, P), $g, r \in \mathbb{Z}_P^*$, which is connected by $g^e \equiv r \pmod{P}$, the probability to find the integer e is negligible.

The DLP forms a new root of assumptions similar to the IFP. The DLP can be formulated in different groups, e.g., $\mathbb{Z}_N$, $\mathbb{Z}_P$ or even *Elliptic Curves*, which makes it one of the most utilized known computational assumptions. The DLP does **not** require the factorization to be unknown, since the DLP can not be solved even if the factors are known. Therefore, it is mostly defined over a prime field that reduces the necessary bit sizes for the involved integers. There are a few cases in which the computation of the discrete logarithm is easy. The Pohlig-Hellman algorithm [99] enables a fast way to compute the DLP in $\mathbb{Z}_P$ if the factorization of $P - 1$ is *smooth*, that means it only consists of small prime factors. The algorithm of Smart [119] shows that special elliptic curves are vulnerable to efficient attacks and in general discrete logarithms are easy to compute in $\mathbb{Z}_{P^2}^*$, if the exponent in question is less than P. For the arbitrary case, the index-calculus algorithm [89] is the most efficient to compute discrete logarithm in $\mathbb{Z}_P$.

Definition 2.2.6 (Computational Diffie-Hellman Assumption (Abbr: C-DHA)) *Let $\mathbb{G}$ be a finite, cyclic subgroup of $\mathbb{Z}_P^*$ with a generator g and a large prime number P. For any probabilistic polynomial time adversary $\mathcal{A}$ who knows the tuple (g^x, g^y) of elements in $\mathbb{G}$, for unknown, random values $x, y \in \mathbb{Z}$, the probability to find the element $g^{xy} \in \mathbb{G}$ is negligible.*

The Diffie-Hellman key agreement scheme is based on the C-DHA. Basically it states that an attacker can not compute the session key if he only eavesdrops the communication and learns the public keys, since he cannot compute discrete logarithms. The C-DHA is a stronger assumption than the DLP (DLP $\Rightarrow$ C-DHA). If the DLP could be solved, the C-DHA is broken since the computation of $\mathrm{dlog}_g(g^x)$ and $\mathrm{dlog}_g(g^y)$ immediately leads to the integer g^{xy}, which is a solution of the C-DHA given (g, g^x, P) and (g, g^y, P). However, it is not clear if an attacker has to solve the discrete logarithm problem in order to solve the C-DHA. In some groups, Maurer and Wolf [82] were able to show that these two problems are indeed polynomial time equivalent.

Definition 2.2.7 (Decisional Diffie-Hellman Assumption (Abbr: D-DHA)) *Let $\mathbb{G}$ be a finite, cyclic subgroup of $\mathbb{Z}_P^*$, whereas P is a prime number of the form $P = 2P'+1$, with P' again being a prime number. Let g be the generator of the group with $J_P(g) = 1$. Given two group elements g^a and g^b of $\mathbb{G}$, for any probabilistic polynomial time ad-*

versary $\mathcal{A}$, the advantage function

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{D-DHA}}(k) = \left| \Pr[\mathcal{A}(p, g, g^{a}, g^{b}, g^{\beta ab+(1-\beta)c}) = \beta] - \frac{1}{2} \right|$$

for random values $a, b, c \in \mathbb{Z}_p$ and random $\beta \in \{0, 1\}$, is a negligible function in k.

The D-DHA is an even stronger assumption than the C-DHA, but it does not hold in all groups the C-DHA is assumed to be true (DLP $\Rightarrow$ C-DHA $\Rightarrow$ D-DHA). The D-DHA's statement is that it is not only infeasible to compute g^{ab} from the group elements g^{a} and g^{b}, but it is even infeasible to decide if a given random integer g^{c} is equal to g^{ab} or not. In other words, it is hard to distinguish the integer g^{ab} from random and an attacker cannot do better than random guessing with a probability of 1/2. However, if someone is able break the C-DHA, he could simply compute g^{ab} and test if it is equal to g^{c} or not, thus he breaks the D-DHA. On the contrary, in some groups the Jacobi-symbol can efficiently be used to decide if elements are equal, which excludes some groups for the D-DHA. The parameters in the definition above show a setup, in which the D-DHA is assumed to be true (P is a Sophie-Germain prime and the group consists of quadratic residues only $\rightarrow J_P(g^{e}) = 1, \forall e \in \mathbb{N}$). Boneh showed more details in his survey about the D-DHA [15].

Definition 2.2.8 (GAP Diffie-Hellman Assumption (Abbr: GAP-DHA)) *Let $\mathbb{G}$ be a finite, cyclic subgroup of $\mathbb{Z}_P^*$ with a generator g and a large prime number P. Given a triple (g, g^{a}, g^{b}) of group elements, find the element g^{ab} with the help of a Decision Diffie-Hellman Oracle.*

The GAP-DHA defines the gap between the computational Diffie-Hellman assumption and the decisional Diffie-Hellman assumption. That means, even an adversary who *can break* the D-DHA, *cannot break* the C-DHA.

2.3 Security Definitions

In this paragraph, properties that are commonly used to characterize a cryptosystem are described.

2.3.1 Encryption Schemes

The first property regarding encryption schemes is called *semantic security* and was first introduced by Goldwasser and Micali [53]. Semantic security offers only a weak level of security, but is an important property that many schemes even do not fulfill. The definition of semantic security is often replaced by an equivalent definition called *ciphertext indistinguishability*, which is described next.

1 : **Indistinguishability under Chosen Plaintext Attack (Abbr: $\mathsf{IND\text{-}CPA}$)**: The property is defined by the following game between a challenger $\mathcal{Cl}$ and the adversary $\mathcal{A}$:

1. $\mathcal{Cl}$ generates a key pair $(\mathsf{public}, \mathsf{private}) = (\mathsf{PK}, \mathsf{SK})$ and sends PK to $\mathcal{A}$ while SK is kept secret.
2. $\mathcal{A}$ uses PK to encrypt a polynomial-bounded number of messages.
3. At any time $\mathcal{A}$ may submit two distinct chosen plaintexts m_0 and m_1 to $\mathcal{Cl}$.
4. $\mathcal{Cl}$ choses $\mathsf{b} \in \{0,1\}$ at random and sends the ciphertext $\mathsf{C} = \mathcal{E}(\mathsf{PK}, \mathsf{m}_\mathsf{b})$ back to $\mathcal{A}$.
5. $\mathcal{A}$ outputs a guess $\mathsf{b}' \in \{0,1\}$ and wins if $\mathsf{b} = \mathsf{b}'$.

The advantage of an $\mathsf{IND\text{-}CPA}$ adversary against $\mathcal{E}$ is the following function:

$$\mathrm{Adv}_{\mathcal{A}}^{\mathcal{E}}(\mathsf{k}) := |\Pr[\mathsf{b} = \mathsf{b}'] - \frac{1}{2}|.$$

Definition 2.3.1 ($\mathsf{IND\text{-}CPA}$ Security) *A system $\mathcal{E}$ is $\mathsf{IND\text{-}CPA}$ secure if for any polynomial time adversary $\mathcal{A}$, the function $\mathrm{Adv}_{\mathcal{A}}^{\mathcal{E}}(\mathsf{k})$ is a negligible function in k.*

The definition says that, if an adversary guesses more often correctly which plaintext the challenger has encrypted than he would do by random guessing (probability of 1/2), the adversary has an advantage about the encryption scheme. In the long run, he could use this advantage to gain information about the encrypted plaintext.

Indistinguishability of chosen plaintext is a mandatory feature for encryption schemes. To see this, imagine a political election that is done with the aid of electronic votingmachines, which use a non IND-CPA secure encryption scheme. Since a ballot paper consists only of a few entries, the number of possible ciphertext is small. An extreme example would be if there were only two values, say ν_1 and ν_2, to be encrypted, one for "'yes"' and one for "'no"'. If an adversary encrypts these two values on its own, using the public key of the voting machine, he can easily compare his results to the intercepted ciphertexts of the electors and thus can decide whether a citizen has voted "'yes"' or "'no"'. An example for a scheme that is **non** IND-CPA is the RSA encryption without padding, since it is deterministic. Whereas RSA with secure padding (e.g. OAEP), which enables IND-CPA security, is until now one of the most utilized schemes in practice. Another scheme that is IND-CPA secure is Paillier's encryption system [94].

Note: A deterministic encryption scheme can never be IND-CPA *secure.* To gain the property of IND-CPA, some kind of randomness has to be involved in the encryption process to make the ciphertext differ each time the same plaintext is encrypted.

A more powerful property is *Indistinguishability under Chosen Ciphertext Attack*. In this case, $\mathcal{A}$ has access to a decryption oracle. (The additional action compared to the previous case is written in italics.)

2 : **Indistinguishability under Chosen Ciphertext Attack (Abbr: IND-CCA):** The property is defined by the following game:

1. $\mathfrak{Cl}$ generates a key pair $(\mathsf{PK}, \mathsf{SK})$ and sends PK to $\mathcal{A}$ while SK is kept secret.
2. $\mathcal{A}$ uses PK to encrypt a polynomial-bounded number of messages *or $\mathcal{A}$ can ask the decryption oracle a polynomially-bounded number of times.*
3. At any time $\mathcal{A}$ may submits two distinct chosen plaintexts m_0 and m_1 to $\mathfrak{Cl}$.
4. $\mathfrak{Cl}$ choses $b \in \{0, 1\}$ at random and sends $C = \mathcal{E}(\mathsf{PK}, m_b)$ back to $\mathcal{A}$.
5. $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins if $b = b'$.

Definition 2.3.2 (IND-CCA Security) *A system $\mathcal{E}$ is* IND-CCA *secure if for any polynomial time adversary $\mathcal{A}$, the function $\mathrm{Adv}_{\mathcal{A}}^{\mathcal{E}}(k)$ is a negligible function in k.*

In this case, the adversary has access to a decryption oracle. The decryption oracle shares the private key of the challenger and responds in a reliable way whenever it is asked for a decryption. Again, the batch RSA encryption system is vulnerable against an $\mathsf{IND\text{-}CCA}$ capable attacker, which is demonstrated next. Therefore, let $\mathcal{A}$ be the attacker who is facing a challenge with some challenger $\mathcal{C}l$. $\mathcal{A}$ chooses two random integers m_0 and m_1 and sends them to the decryption oracle, that responses with $M_0 \equiv m_0^d \pmod N$ and $M_1 \equiv m_1^d \pmod N$. Next, $\mathcal{A}$ sends the two received answers (M_0, M_1) to the challenger. According to the rules of the game, $\mathcal{C}l$ select one of them randomly and computes $M_i^e \equiv c_i \pmod N$, which he sends back to $\mathcal{A}$. To decide which message $\mathcal{C}l$ chose for his encryption, $\mathcal{A}$ simply compares c_i with m_0 and m_1, since $c_i \equiv M_i^e \equiv m_i^{ed} \equiv m_i \pmod N$. Thus attacker easily wins the game with perfect probability.

Next, an even stronger case of security property is shown. It is the demanded security level for an encryption system used in practice.

3. **Indistinguishability under Adaptive Chosen Ciphertext Attack (Abbr: $\mathsf{IND\text{-}CCA2}$):** The property is defined by the following game:

1. $\mathcal{C}l$ generates a key pair (PK, SK) and sends PK to $\mathcal{A}$ while SK is kept secret.
2. $\mathcal{A}$ uses PK to encrypt a polynomial-bounded number of messages *or $\mathcal{A}$ can ask the decryption oracle a polynomially-bounded number of times.*
3. At any time $\mathcal{A}$ may submits two distinct chosen plain-texts m_0 and m_1 to $\mathcal{C}l$.
4. $\mathcal{C}l$ choses $b \in \{0, 1\}$ at random and sends $C = \mathcal{E}(PK, m_b)$ back to $\mathcal{A}$.
5. *$\mathcal{A}$ can query the decryption oracle a polynomially-bounded number of times (but not ask for the decryption of C).*
6. $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins if $b = b'$.

Definition 2.3.3 **($\mathsf{IND\text{-}CCA2}$ Security)** *A system $\mathcal{E}$ is $\mathsf{IND\text{-}CCA2}$ secure if for any polynomial time adversary $\mathcal{A}$, the function $\mathrm{Adv}_{\mathcal{A}}^{\mathcal{E}}(k)$ is a negligible function in k.*

In this case, the attacker cannot only ask the decryption oracle before submitting his two plaintexts to $\mathcal{C}l$, but even after he receives the challenge. Obviously, he is not allowed to ask for the decryption of the challenge directly, which would immediately reveal the used message. But he is allowed to ask for any object, which could even have emerged from the challenge after it has been modified in an arbitrary way. Since RSA is already non $\mathsf{IND\text{-}CPA}$ and non $\mathsf{IND\text{-}CCA}$ secure, it can not have the property of $\mathsf{IND\text{-}CCA2}$. For completeness it will be shown how an attacker could use the decryption queries to break the batch RSA scheme. Let $C_i = m_i^e \pmod N$ be a received ciphertext $\mathcal{A}$ got from the challenger $\mathcal{Cl}$. Now, the attacker $\mathcal{A}$ computes $C_i' \equiv C_i 2^e \pmod N$ and asks the decryption oracle for a decryption of C_i'. Note that $\mathcal{A}$ does not ask for the decryption of C_i itself, which he is not allowed to do, but for the decryption of the product $C_i 2^e$, which is a random element in $\mathbb{Z}_N$. As the result $\mathcal{A}$ receives $2m_i$, whereof he can easily derive m_i and wins the challenge. A cryptosystem that is $\mathsf{IND\text{-}CCA2}$ secure is the Cramer-Shoup encryption system [39].

Malleability. Another property of a cryptosystem is *malleability*. Informally, an encryption algorithm is malleable if it is possible for an adversary to transform a ciphertext $E(m) = c$ of a message m into another ciphertext c' such that c' decrypts to a desired plaintext m'. A cryptosystem that is malleable should, for example, not be used for money transactions or bidding auctions. Again, RSA used without a padding scheme is malleable, as shown by a little example: In an online auction, $\mathcal{A}$ offers an object for a certain start amount. A participant $\mathcal{B}$ encrypts its bidding value v and sends it as a ciphertext c to $\mathcal{A}$. Assume now a competitor $\mathcal{K}$ that also is interested in the object offered by $\mathcal{A}$. But instead of bidding more than competitor $\mathcal{B}$, he decides to let $\mathcal{B}$ bid less. Therefore, $\mathcal{K}$ intercepts and manipulates $\mathcal{B}$'s encrypted value c by computing $ct^e \equiv (vt)^e \equiv c' \pmod n$. Thus, $\mathcal{K}$ changes the original bidding amount by a factor of t, since c' decrypts to tv instead of v.
On the other hand, some schemes are designed to be malleable and it is a mandatory feature for some applications. For example, homomorphic schemes that are used for encrypted computations (doing computations on encrypted data) are designed exactly for this purpose. A cryptosystem which is $\mathsf{IND\text{-}CPA}$ and $\mathsf{IND\text{-}CCA}$ secure can still be malleable, whereas a $\mathsf{IND\text{-}CCA2}$ secure system is non-malleable per definition.

2.3.2 Signatures

Next, a definition that is related to the security of an identity based signature scheme is given. For such a scheme, it is important that an adversary cannot issue a signature for an identity, for which the adversary does not have the corresponding private key.

1. A ID-based signature scheme is said to be secure against **existential forgery on adaptively chosen message and ID attacks** (Abbr: $\mathsf{ID\text{-}ACM}$) if no polynomial time bounded adversary $\mathcal{A}$ has a non-negligible advantage against a challenger $\mathfrak{Cl}$ in the following game:

1. $\mathfrak{Cl}$ creates the public shared parameters for the ID scheme and returns them back to $\mathcal{A}$.

2. $\mathcal{A}$ can make the following queries to $\mathfrak{Cl}$:

   a) A Hash function query: $\mathcal{A}$ queries $\mathfrak{Cl}$ for the hash value of a given input (message or identity). $\mathfrak{Cl}$.

   b) Extract query: $\mathcal{A}$ can ask $\mathfrak{Cl}$ the private key for an arbitrary identity from $\mathfrak{Cl}$

   c) Sign query: $\mathcal{A}$ request $\mathfrak{Cl}$ for the signature on $\mathfrak{m}$ issued by the identity ID

3. $\mathcal{A}$ outputs (ID,m,s) as a forgery, where ID was not part of an Extract query and (m,ID) not part of a Sign query before. $\mathcal{A}$ wins if the signature $\mathfrak{s}$ is a valid signature on $\mathfrak{m}$ generated by ID.

Since the attacker has ”‘adaptive”’ skills, it is the most secure definition an ID-based signature scheme can fulfill.

Definition 2.3.4 ($\mathsf{ID\text{-}ACM}$ Security) *An ID-signature scheme $\mathcal{E}$ is* $\mathsf{ID\text{-}ACM}$ *secure if for any polynomial time adversary $\mathcal{A}$ the probability to forge a signature is negligible.*

2.3.3 Key Agreement Protocols

Since in a key agreement protocol no ciphertext occurs, the definitions from the previous section can not be applied. The goal of a key agreement protocol is to let two users negotiate a session key, whereas both users contribute to the key in an equal share. This stays in contrast to a key exchange protocol, where only one participant determines the session key. Thus, a key agreement scheme has per definition a higher level of security than a key exchange protocol, but needs at least one more communication step for completion.

Several definitions for a secure key agreement protocol were given in the literature. Here, the Canetti-Krawczyk Model [30, 31] is used. An abridged description of the model is presented below. For more information the reader in referred to original papers.

Canetti-Krawczyk Model (CKM). In the CKM, a key agreement protocol is executed in a network of interconnected peers, which can run an instance of the protocol, called a *session*. According to the specification of the protocol, a participant creates and maintains a *session state*. He creates outgoing messages, receives incoming messages and eventually *completes* the session by outputting a *session key* at the end. If $\mathsf{E}_{\mathsf{ID}1}$ and $\mathsf{E}_{\mathsf{ID}2}$ initiate a session among each other, a session identifier stored by the $\mathsf{E}_{\mathsf{ID}1}$ is of the form $(\mathsf{E}_{\mathsf{ID}1}, \mathsf{E}_{\mathsf{ID}2}, \mathsf{In}_1, \mathsf{Out}_1)$, where $\mathsf{E}_{\mathsf{ID}2}$ has $(\mathsf{E}_{\mathsf{ID}2}, \mathsf{E}_{\mathsf{ID}1}, \mathsf{In}_2, \mathsf{Out}_2)$. If $\mathsf{In}_1 = \mathsf{Out}_1$ and $\mathsf{In}_2 = \mathsf{Out}_1$, then the two sessions are *matching*.

An attacker is allowed to make the following queries to each participant, which are answered honestly:

State-Reveal Query: This query is directed at a single, incomplete session and he receives the session state for that single session.

Session-Key Query: The query is done on a completed session, which is responded with the session key for this session.

Party Corruption Query: Using this query, the attacker learns all information from the participant, that means all session keys/states and even the private identity key. After a Party Corruption Query the affected participant is completely controlled by the attacker.

Whenever a session is faced with one of the above queries, it is called **exposed**. Next, a game is defined that an adversary has to win with an advantage over the long run to be a successful key agreement attacker. Therefore, a simulator Sim simulates all participants and messages. He executes arbitrary protocol runs between random parties. Whenever he receives one of the above defined queries from the attacker, he answers honestly. The goal of the attacker is to distinguish a session key of an unexposed session from a random element.

1. Sim generates the key agreement world with all participants and keys and executes the protocol π between random peers.

2. Whenever the attacker $\mathcal{A}$ decides to query a **State Reveal Query**, a **Session Key Query** or a **Party Corruption Query** he does so and gets answered honestly.

3. At any time, $\mathcal{A}$ can decide to chose a *test session* that must be unexposed and completed and notifies the simulator for a challenge.

4. Sim tosses a coin b, $\mathsf{b} \in_{\mathsf{R}} \{0, 1\}$. If $\mathsf{b} = 0$, he submits the real session key of the test session to $\mathcal{A}$, else he submits a random value.

5. $\mathcal{A}$ can continue his queries to get more information, but he is not allowed to ask queries on the chosen test session.

6. At the end, $\mathcal{A}$ outputs a guess b' and wins if $\mathsf{b} = \mathsf{b}'$.

Definition 2.3.5 (Authenticated Key Agreement Protocol) *A polynomial-time attacker with the capabilities to make **State Reveal Queries**, **Session Key Queries** and **Party Corruption Queries** is called a key agreement attacker. A key agreement protocol π is called secure if for all key agreement attackers running against π the following holds:*

1. *If two uncorrupted parties complete matching sessions in a run of protocol π, then the session key output in these sessions is the same; and*

2. *the probability of the advantage function*

$$\mathrm{Adv}^{\pi}_{\mathcal{A}}(\mathsf{k}) := \left| \Pr[\mathsf{b} = \mathsf{b}'] - \frac{1}{2} \right|$$

is a negligible function in k.

Some attack scenarios are not covered by the definition above and have to be defined separately.

Perfect Forward Secrecy (PFS). A key agreement protocol is said to be secure with perfect forward secrecy if the compromise of the private identity key does not compromise past session keys.

Key Compromise Impersonation (KCI). If an attacker obtains the private identity key of an user E_{ID}, he is able to impersonate this user towards other members. However, it should not be possible for an attacker to impersonate other identities to E_{ID}.

Unknown Key-Share (UKS). An entity $\mathsf{E}_{\mathsf{ID}1}$ cannot be coerced into sharing a key with entity $\mathsf{E}_{\mathsf{ID}2}$ without $\mathsf{E}_{\mathsf{ID}1}$'s knowledge. That is, when $\mathsf{E}_{\mathsf{ID}1}$ believes the key is shared with some entity $\mathsf{E}_{\mathsf{ID}3} \neq \mathsf{E}_{\mathsf{ID}2}$ and $\mathsf{E}_{\mathsf{ID}2}$ (correctly) believes the key is shared with $\mathsf{E}_{\mathsf{ID}1}$.

2.4 The Standard and the Random Oracle Model

To prove the security of a cryptographic scheme, one way is to show that the security can be reduced to a problem that is already known to be hard (i.e. the problems introduced in Paragraph 2.2). This means, it must be shown that any algorithm that breaks the cryptographic scheme in question, can also be utilized to break the mathematical problem the scheme is based on. This kind of proofs are called "'reductions"' and have often been used successfully to prove certain protocols.

There are two main models that are used for this purpose, first the *Standard Model* and second the *Random Oracle Model* [12]. Proofs in the Standard Model only use assumptions about the time and space complexity which is necessary to solve the problem. Even though this sounds easy, if already the underlying problem does need exponential time and space complexity, it is often difficult to proof a scheme in the Standard Model. Just in the case when some kind of randomness is involved, which is essential for a protocol to obtain the property of *semantic security*, a proof cannot be formulated in the Standard Model easily. In this case, proofs take advantage of the Random Oracle Model. In this model it is assumed that there is an instance (the Random Oracle) that responds to every question with a truly random answer. The distribution of its answers should be uniformly distributed from its output domain. Every time the oracle gets the same input, it outputs the same answer as in the previous case. The Random Oracle gives the prover a way to handle randomness by using the random model as its source. The field of application for random oracles is to model mathematical abstraction from objects which have no existing instantiations in real-life, like for example cryptographic hash-functions. Random Oracles do not exist in real life, thus proofs relying on an artificial object seem on the first sight disappointing. Even worse, in 2004 Canetti et. al published a paper [28] that shows that there are protocols that can be proven secure in the random oracle model but will result in an insecure protocol whenever the random oracle is replaced by an existing function. However, an attacker will only succeed if he demands unrealistic behavior from the random oracle, which is accepted to be sufficient for a protocol to be secure.

There are two ways a random oracle is used in protocols. The first is to allow the attacker free access to the random oracle which responds with random and non further specified values. In the other case, another algorithm *simulates* the random oracle by answering with certain values, which can not be distinguished from a random distribution but are meaningful for the oracle. These are called *Programmable Random Oracles*. Mostly it is shown that if an attacker succeeds in breaking the protocol in question, the involved programmed random oracle can use its generated answers as well as the attacker as a subroutine and to break a certain assumption.

2.5 Summary

In this chapter, several notations and statements that are used during the rest of the thesis were defined. The fundamental assumptions that are used in cryptography were presented. Especially notable is the PHA, which is investigated in further detail in Chapter 5. Security definitions for the three classes *Encryption Scheme*, *Signature* and *Key Agreement* were given. The latter two will be used to prove the security of the scheme presented in this thesis. Finally, a short overview of the Standard- and the Random Oracle was given.

3 Secure Session Framework

"A good scientific theory should be explicable to a barmaid."
Ernest Rutherford

3.1 Introduction

In this chapter, the main contribution of this thesis is presented, namely the *Secure Session Framework* (SSF). The chapter begins with the algorithms of the basic key agreement scheme. Afterwards, the scheme is extended to support multiple independent key generators. This novel extension method makes SSF the fist identity-based key agreement scheme in the literature that supports such generators. The later part of this chapter describes the algorithms that turn SSF into a multi-signature scheme. By overtaking the methods that enable multiple generators in the key agreement case to the signature case, SSF is the first framework that includes multi-signatures from multiple independent ID-PKGs.

The main protocol has been published in [108]; the signature part has been published in [109, 107]. Finally, four patent applications that contain the main parts of the scheme have been filed [113, 112, 114, 111].

3.2 Key Agreement

Since the first mention of identity-based cryptography in 1984, several schemes have been proposed. However, it took several years before a scheme was found that fulfills the requirements for a secure system. In 2001, Cocks proposed a promising encryption system [35] based on quadratic residues. Although it could achieve the high level of IND-

CCA2 security, the scheme is inefficient in practice because of its enormous ciphertext extension. That is, each bit of the plaintext is enlarged to the bit length of the entire modulus, which is a factor of about > 1000[1]. In 2003, Boneh and Franklin published an identity-based encryption scheme [22] that is both IND-CCA2 secure and efficient in practice. As a key approach, they utilized bilinear pairings on elliptic curves. The usage of pairings was not new in cryptography and was already introduced in 2001 by Menezes et. al [85] to reduce discrete logarithms from elliptic curves to finite fields. However, the application to identity-based cryptography was a great idea to make it work in practice. Since their publication, a multiplicity of identity-based systems were proposed that either used the idea of using bilinear pairings [29, 62, 126, 21] or picked up the idea of Cocks [20, 21].

The first key agreement protocols based on identity-based cryptography can be dated back to the eighties of the 20-th century [116], [92], [93], [57], [83]. At later time, also bilinear pairings were applied to key agreement protocols [32], [4], [117].

The most basic and efficient key agreement protocol known is the plain Diffie-Hellman protocol [40]. It needs only two messages, one for each party, to agree on a common integer. Furthermore, it gets along with only two exponentiations and is based on a well reviewed number theoretic problem (the DLP). Even if it has come into ages, it is still utilized in a large number of applications because of its simplicity and security. However, the protocol completely misses key authentication and is thus vulnerable to MITM attacks.

An identity-based scheme with the same complexity, but with authentication based on the involved identity, is the Okamoto-Tanaka protocol (OkTa) [92]. Unfortunately, the OkTa protocol is vulnerable with respect to several attacks, as shown later in the thesis.

One of the open problem in the IBC literature is to create a way to handle multiple identity private key generators (ID-PKGs), hosted by different authorities, which can operate completely independent of each other. The problem that occurs if users from different ID-PKGs try to establish a key agreement is that the involved private keys do only match to their identity regarding the set of public shared parameters that belong to their assigned ID-PKG. With respect to the foreign set of public shared parameters, the identity key and the identity do not have any relationship, making a key agreement impossible. Furthermore, a solution should require only a minimum effort for the involved parties. Neither the ID-PKGs nor the users should be burdened with

[1]Based on the fact that a modern secure system should at least use a modulus of size $> 2^{1024}$.

additional communication or time consuming computation. Several solutions have been proposed that allow multiple ID-PKGs to interoperate ([32, 64, 84, 24, 16]), but these systems require either cooperation between the ID-PKGs or a hierarchical approach with a trusted party at the top. Both of these approaches are difficult to use, e.g., in the Internet, due to organizational difficulties and conflicting business interests. As demonstrated by approaches based on a Certificate Authority (CA), there will always be competing organizations offering the same service for the same protocol (e.g. signing RSA public keys) without wanting to cooperate on the corporate level.

Before the protocol and some related work is presented, notations and algorithms that are commonly used in the area of IBC are introduced in order to simplify the description of the schemes.

Notations and Protocol Overview. Whenever an entity (sometimes called identity) is mentioned, the actual communication unit is meant, e.g. a person or a computer. This entity is denoted with E. For an identifier (identity string), the symbol ID is used. Finally, a specific entity E with an identifier ID is denoted as E_{ID}.

An identity-based key agreement protocol consists of the four algorithms: Setup, Extract, BuildSIK (SIK = Session Initiation Key) and Compute. Next, the four algorithms are first described in a short and informal way to illustrate their functionality:

Setup: Given two bit length parameters k_1 and k_2, the Setup algorithm, executed by the ID-PKG, generates a set of public shared parameters (PSP) and the corresponding secret parameters (SP).

Extract: The Extract algorithm, executed by the ID-PKG, receives the PSP, SP and an identity string $\mathsf{ID} \in \{0,1\}^*$ of an entity E_{ID} as its input. The output is an identity key d_{ID}, which is E_{ID}'s identity key (or often called longterm secret key).

BuildSIK: The BuildSIK algorithm is executed by the entities performing the key agreement and builds a Session Initiation Key, using the private identity key d_{ID}, a random integer r and the PSP as its input. Each party in the key agreement generates a SIK and sends it to the other party. The SIK can be interpreted as a temporary public key used for a single key agreement.

Compute: The Compute algorithm is executed by the parties performing the key agreement and takes a received SIK from the key agreement partner E_{ID}, the random private integer r from its own BuildSIK step, the remote identity string ID and the PSP as its inputs. The output is a common integer $\mathsf{S} \in \mathbb{Z}_\mathsf{N}$ shared by both key agreement parties, which can be used as the input for a key derivation function to obtain the final shared session key.

The four algorithms are all related to the case when all involved entities got keys from the same ID-PKG. For a more generic case, the protocol also has to handle the case when each or some of the entities got keys from different ID-PKGs.

Multiple ID-PKGs. In this case there are three algorithms, the extend algorithm Extend, the build extended SIK algorithm $\mathsf{BuildSIK}_\mathsf{MultIDPKG}$ and also a compute algorithm $\mathsf{Compute}_\mathsf{MultIDPKG}$. Again in an informal way, the three algorithm are introduced next:

Extend: Given PSP$_1$, PSP$_2$, d_ID and the remote identity ID as its input, the Extend algorithm extends the identity key d_ID to a new identity key $\widetilde{\mathsf{d}}_\mathsf{ID}$ in $\mathbb{Z}_{\mathsf{N}_1\mathsf{N}_2}$. It does the same with the hash values of the remote $\mathsf{H}(\mathsf{ID})$ that is extended to a new integer $\widetilde{\mathsf{H}}(\mathsf{ID})$ in $\mathbb{Z}_{\mathsf{N}_1\mathsf{N}_2}$. The algorithm is executed by each E_ID and does not need any SPs.

$\mathsf{BuildSIK}_\mathsf{MultIDPKG}$: Given PSP$_1$, PSP$_2$, $\widetilde{\mathsf{d}}_\mathsf{ID}$ and a random integer r as its input, this algorithm computes an extended session initiation key (eSIK) in $\mathbb{Z}_{\mathsf{N}_1\mathsf{N}_2}$. The algorithm is executed by each E_ID and does not need any SPs.

$\mathsf{Compute}_\mathsf{MultIDPKG}$: The algorithm takes the eSIK from a communication partner E_ID of a remote domain, the random private integer r, the extended hash value of the remote identity $\widetilde{\mathsf{H}}(\mathsf{ID})$ and the PSP$_1$ and PSP$_2$ as its inputs. The output is a common integer $\mathsf{S} \in \mathbb{Z}_{\mathsf{N}_1\mathsf{N}_2}$ shared by both key agreement parties, which can be used as the input for a key derivation function.

3.3 Related Protocols

Before the main scheme is introduced, the identity-based key agreement protocols that operate under similar circumstances are briefly reviewed.

3.3.1 Maurer-Yacobi

The identity key agreement protocol of Maurer and Yacobi [83] is based one a certain setting that allows to compute discrete logarithms in composite rings. To make this possible, they run the following setup algorithm:

Algorithm 1 (Mau/Yac) Setup

```
Input: $k, r$
Output: {PSP, SP}
1. $\{p_1, p_2, ..., p_r\} \overset{\$}{\leftarrow} \mathcal{PRIMES}(k)$
2. $N = \prod_{i=1}^{r} p_i$
3. Choose an integer $G$ that is primitive in every $\mathbb{F}_{p_i}$, $i = 1, ...r$
4. return $\{(N, G), (p_1, p_2, ..., p_r)\}$
```

The public modulus N in their protocol is the product of r distinct prime numbers. To avoid easy factoring, the bit length of each of these primes must be sufficiently large. For private key extraction Algorithm 2 is utilized.

Algorithm 2 (Mau/Yac) Extract

```
Input: PSP, SP, $ID$
Output: private identity key $d_{ID}$
1. $d_{ID} \equiv \log_G(ID^2) \pmod{N}$
2. return $d_{ID}$
```

As it can be seen, the identity key d_{ID} is the discrete logarithm of the squared identity. Computing discrete logarithms is actually a one-way function and should be infeasible even if the factorization of the modulus is known. However, if the involved prime numbers are moderate in size, the known discrete logarithm algorithms become tolerably feasible on powerful computers. Thus, the protocol makes a dangerous trade-off between choosing primes that are large enough to resist factoring, but small enough the let discrete logarithms be feasible to compute.

If the ID-PKG operator chooses a modulus N, that is known to be hard to factorize, like an RSA modulus with $N = pq$ (thus $r = 2$) and p and q primes both of size ≈ 1024 bit, the ID-PKG will **not** be able to compute discrete logarithms in a lifetime. Reducing the size and increasing r eases the work of factoring but enables to compute the necessary logarithms. A parameter analysis of the Maurer-Yacobi scheme from 2006 [71] shows that the scheme is hardly practical at all.

3.3.2 Okamoto-Tanaka

The protocol of Okamoto and Tanaka [92] operates in RSA rings and has the following setup:

Algorithm 3 (Ok/Ta) Setup

```
Input: $k_1$,$k_2$
Output: {PSP, SP}
1. $R \xleftarrow{\$} \{2^{k_1-1}, 2^{k_1}-1\}_{odd}$
2. $P \xleftarrow{\$} \mathcal{PRIMES}(k_2)$
3. if $R|(P-1)$ goto 2
4. $Q \xleftarrow{\$} \mathcal{PRIMES}(k_2)$
5. if $R|(Q-1)$ goto 4
6. $N \leftarrow P \cdot Q$
7. $G \xleftarrow{\$} \mathbb{Z}_N^*$
8. return $\{(N, G, R), (P, Q)\}$
```

This is a usual setup for protocols that are based on the IFP. The Setup algorithm does not utilize a hash function and neither does the Extract algorithm:

Algorithm 4 (Ok/Ta) Extract

```
Input: PSP, SP, ID
Output: $d_{ID}$
1. $d_{ID} \equiv ID^{1/R} \pmod N$
2. return $d_{ID}$
```

The flaw of omitting the hash function step makes the protocol vulnerable to several attacks:

1. Suppose the ID of a participant is *not* hashed before the private key is extracted out of it. In such a system, an adversary $\mathcal{A}$ is able to reveal the private key of *any* user. Therefore, the adversary selects an arbitrary user to attack, say the user with identity ID_1. Afterwards, $\mathcal{A}$ sets its own identity to $ID_{\mathcal{A}} \equiv ID_1 r^R \pmod N$, for some random value $r \in \mathbb{Z}_N$. The adversary requests its private key from the key generator and receives

$$d_{ID_{\mathcal{A}}} \equiv (ID_1 r^R)^{1/R} \equiv d_{ID_1} r \pmod N.$$

After dividing out r, $\mathcal{A}$ is in the possession of the private key for ID_1. On the contrary, a protocol that uses hashing is prevented from this attack, since the adversary only gets the private key for $H(ID_1 r^R)$, which is non-related to $H(ID_1)$.

2. If the identity is not hashed, *Unknown Key Share* (UKS) attacks are possible. Again, the adversary $\mathcal{A}$ choses ID_1 to attack. In this case $\mathcal{A}$ sets its own identity to $\mathsf{ID}_{\mathcal{A}} \equiv \mathsf{ID}_1\mathsf{G}^{\mathsf{tR}}$ (mod N), with t a random integer, and $\mathsf{G}, \mathsf{R}, \mathsf{N}$ the usual system parameters. During the key agreement of $\mathsf{E}_{\mathsf{ID}1}$ and $\mathsf{E}_{\mathsf{ID}2}$, $\mathcal{A}$ intercepts the initiating packet $\mathbf{p} = \mathsf{G}^{\mathsf{r}_1}\mathsf{d}_{\mathsf{ID}_1}$ of $\mathsf{E}_{\mathsf{ID}1}$ and replaces it with $\mathbf{p}\mathsf{G}^{\mathsf{t}}$ and correctly marking it as a packet coming from $\mathcal{A}$. $\mathsf{E}_{\mathsf{ID}2}$ answers honestly with a message $\mathsf{G}^{\mathsf{r}_2}\mathsf{d}_{\mathsf{ID}_2}$ and $\mathcal{A}$ simply relays the packet unchanged to $\mathsf{E}_{\mathsf{ID}1}$. Now $\mathsf{E}_{\mathsf{ID}1}$ computes

$$((\mathsf{G}^{\mathsf{r}_2}\mathsf{d}_{\mathsf{ID}_2})^{\mathsf{R}}\mathsf{E}_{\mathsf{ID}2}^{-1})^{2\mathsf{r}_1} \equiv \mathsf{G}^{2\mathsf{R}\mathsf{r}_1\mathsf{r}_2} \pmod{\mathsf{N}}$$

while $\mathsf{E}_{\mathsf{ID}2}$ computes

$$(\mathbf{p}^{\mathsf{R}}\mathsf{ID}_{\mathcal{A}}^{-1})^{2\mathsf{r}_2} \equiv (\mathsf{G}^{\mathsf{r}_1}\mathsf{d}_{\mathsf{ID}_1}\mathsf{G}^{\mathsf{t}})^{\mathsf{R}}\mathsf{ID}_1\mathsf{G}^{-\mathsf{tR}})^{2\mathsf{r}_2} \equiv \mathsf{G}^{2\mathsf{R}\mathsf{r}_1\mathsf{r}_2} \pmod{\mathsf{N}}$$

The session key is the same in both computations, but $\mathsf{E}_{\mathsf{ID}1}$ associates the key with $\mathsf{E}_{\mathsf{ID}2}$, while $\mathsf{E}_{\mathsf{ID}2}$ connects it with $\mathcal{A}$.

3.4 SSF Key Agreement

3.4.1 With Single ID-PKG

In the sequel, the basic key agreement protocol proposed in this thesis is presented. It comes with the same cost as the plain Diffie-Hellman protocol, but it does not suffer from the same vulnerabilities as the Okamoto-Tanaka protocol.

Notation: All integers that are part of the public, shared parameters are written in capital letters to distinguish them from the user based variables that are written in small letters.

The first step is the Setup algorithm that is executed by the ID-PKG. This algorithm is only executed once to create the secret parameters (SP) and the corresponding public parameters (PSP).

The algorithm uses two security parameters k_1 and k_2 that determine the bit length of two involved integers. The first parameter k_1 can be rather small, e.g, 2 or 3. It specifies the range of the integer R, which is equal to the public integer e in a RSA encryption and has been proven secure also for small values. The second parameter k_2

Algorithm 5 (SSF) Setup

```
Input: $k_1$,$k_2$
Output: {PSP, SP}
1. $R \xleftarrow{\$} \{2^{k_1-1}, 2^{k_1}-1\}_{odd}$
2. $P \xleftarrow{\$} \mathcal{PRIMES}(k_2)$
3. if $R|(P-1)$ goto 2
4. if $(P-1)/2$ is not prime goto 2
5. $Q \xleftarrow{\$} \mathcal{PRIMES}(k_2)$
6. if $R|(Q-1)$ goto 4
7. if $(Q-1)/2$ is not prime goto 4
8. $N \leftarrow P \cdot Q$
9. $G \xleftarrow{\$} \mathbb{Z}_N^*$
10. $H(\cdot) \leftarrow$ collision-resistant hash function that maps into the group generated by $G$.
11. return $\{(N, G, R, H), (P, Q)\}$
```

determines the bit length of the involved prime numbers that make up the modulus N. Regarding modern factorization algorithms and powerful hardware, k_2 should chosen sufficiently large. The algorithm checks whether $P-1$ as well as $Q-1$ is co-prime to R and additionally whether the two primes are *Sophie-Germain-Primes* (**Def**: A prime number p is called a Sophie-Germain-Prime if the integer $(p-1)/2$ is also a prime number). Such primes are often utilized in cryptography since they posses only a few of subgroups in $\mathbb{Z}_p$, which countermeasures some factoring approaches.

Based on Algorithm 5, the public shared parameters as well as the private parameters are defined as:

Public, Shared Parameters. *The public, shared parameters* (PSP) *of the key agreement scheme* SSF *is the quadruple* PSP := (N, G, R, H)

Secret ID-PKG Parameters. *The secret parameters* (SP) *of a ID-PKG of the key agreement scheme* SSF *is the tuple* SP := (P, Q).

The Extract algorithm creates the identity key for a given identity string. This algorithm is executed by the ID-PKG. Note that the hash operation is applied before extracting the roots.

The Extract algorithm actually describes a one-way function (see RSAA) and can only be inverted knowing the secret parameters P and Q.

The BuildSIK algorithm is executed by the parties performing the key agreement. The

Algorithm 6 (SSF) Extract

```
Input: PSP, SP, $\mathsf{ID}$
Output: $d_{\mathsf{ID}}$
1. $d_{\mathsf{ID}} \equiv \mathsf{H}(\mathsf{ID})^{1/R} \pmod{N}$
2. return $d_{\mathsf{ID}}$
```

SIK is in fact the product of the user's private identity key and a Diffie-Hellman key. The SIKs can be exchanged over an unsecured channel, since the owner's identity is involved, which makes them *invulnerable* to a MITMA as long as the identity is known.

Algorithm 7 (SSF) BuildSIK

```
Input: PSP, $d_{\mathsf{ID}}$, $k$
Output: $\mathrm{SIK}_{\mathsf{ID}}$
1. $r \xleftarrow{\$} \{2^{k-1}, 2^k - 1\}$
2. $\mathrm{SIK}_{\mathsf{ID}} \equiv G^r \cdot d_{\mathsf{ID}} \pmod{N}$
3. return $\mathrm{SIK}_{\mathsf{ID}}$
```

The final step of the key agreement process is the computation of the session key using the Compute algorithm that is executed by the participants performing the key agreement.

Algorithm 8 (SSF) Compute

```
Input for $E_{\mathsf{ID}_1}$: $\mathsf{ID}_2$, PSP, $\mathrm{SIK}_{\mathsf{ID}_2}$, $r_{\mathsf{ID}_1}$
Input for $E_{\mathsf{ID}_2}$: $\mathsf{ID}_1$, PSP, $\mathrm{SIK}_{\mathsf{ID}_1}$, $r_{\mathsf{ID}_2}$
Output: common integer $S$
1. $(\mathrm{SIK}_2^R \cdot \mathsf{H}(\mathsf{ID}_2)^{-1})^{2r_{\mathsf{ID}_1}} \equiv ((G^{r_{\mathsf{ID}_2}} \cdot d_{\mathsf{ID}_2})^R \cdot \mathsf{H}(\mathsf{ID}_2)^{-1})^{r_{\mathsf{ID}_1}} \equiv G^{2Rr_{\mathsf{ID}_1}r_{\mathsf{ID}_2}} \equiv S \pmod{N}$
2. $(\mathrm{SIK}_1^R \cdot \mathsf{H}(\mathsf{ID}_1)^{-1})^{2r_{\mathsf{ID}_2}} \equiv ((G^{r_{\mathsf{ID}_1}} \cdot d_{\mathsf{ID}_1})^R \cdot \mathsf{H}(\mathsf{ID}_1)^{-1})^{r_{\mathsf{ID}_2}} \equiv G^{2Rr_{\mathsf{ID}_1}r_{\mathsf{ID}_2}} \equiv S \pmod{N}$
return $S$
```

The output is a common integer S shared between E_{ID_1} and E_{ID_2}. To derive a symmetric session key, e.g., for an AES encryption, a key derivation function can be used (e.g. SHA-256).

Lemma 3.4.1 *The* SSF *key agreement algorithm is correct.*

Proof 3.4.2 (of Lemma 3.4.1) *The proof follows directly from the definition of the* Compute *algorithm.*

Protocol Flow. Figure 3.1 illustrates the actual protocol flow. It is assumed that both participants possess the same set of public shared parameters. In the first step, $E_{\mathsf{ID}1}$ computes its session initiation key using Algorithm 7. Subsequent, $E_{\mathsf{ID}1}$ sends the generated SIK to $\mathsf{E}_{\mathsf{ID}2}$ describing the identity ID_1 as the packet's source. In the third step, $\mathsf{E}_{\mathsf{ID}2}$ generates a SIK on his part, which he sends to $\mathsf{E}_{\mathsf{ID}1}$ in step four, again, appending his identity as the source. ($\mathsf{x}|\mathsf{y}$ *is used if packet* y *is send using identity* x*. E.g., assume* x *is the source address in an IP packet.*)

Setting: Both participants possess the PSP $= (\mathsf{N}, \mathsf{G}, \mathsf{R}, \mathsf{H})$

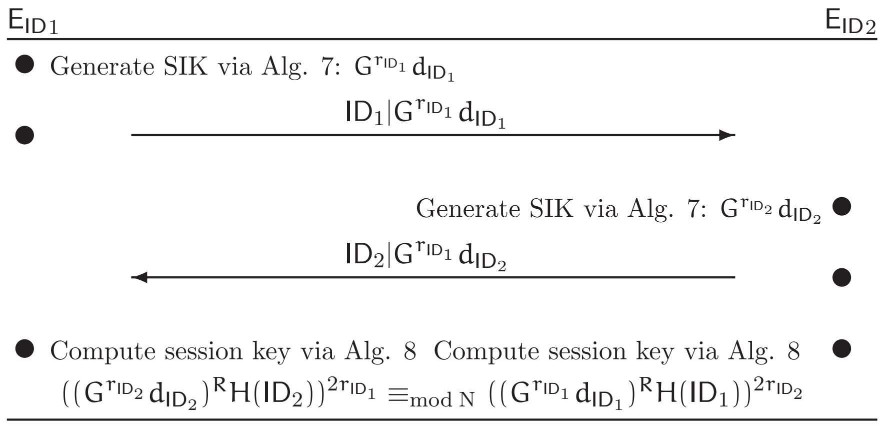

$\mathsf{E}_{\mathsf{ID}1}$
$\mathsf{E}_{\mathsf{ID}2}$
Generate SIK via Alg. 7: $\mathsf{G}^{\mathsf{r}_{\mathsf{ID}1}}\mathsf{d}_{\mathsf{ID}_1}$
$\mathsf{ID}_1|\mathsf{G}^{\mathsf{r}_{\mathsf{ID}1}}\mathsf{d}_{\mathsf{ID}_1}$
Generate SIK via Alg. 7: $\mathsf{G}^{\mathsf{r}_{\mathsf{ID}2}}\mathsf{d}_{\mathsf{ID}_2}$
$\mathsf{ID}_2|\mathsf{G}^{\mathsf{r}_{\mathsf{ID}1}}\mathsf{d}_{\mathsf{ID}_2}$
Compute session key via Alg. 8
Compute session key via Alg. 8
$((\mathsf{G}^{\mathsf{r}_{\mathsf{ID}2}}\mathsf{d}_{\mathsf{ID}_2})^{\mathsf{R}}\mathsf{H}(\mathsf{ID}_2))^{2\mathsf{r}_{\mathsf{ID}1}} \equiv_{\text{mod N}} ((\mathsf{G}^{\mathsf{r}_{\mathsf{ID}1}}\mathsf{d}_{\mathsf{ID}_1})^{\mathsf{R}}\mathsf{H}(\mathsf{ID}_1))^{2\mathsf{r}_{\mathsf{ID}2}}$

Figure 3.1: The SSF protocol flow.

In the last step, both participants use the received SIK and execute Algorithm 8 to compute the integer S.

Communication and Computational Cost. The efficiency of the protocol, as can be seen from the Compute algorithm, regarding computational costs and communication steps is the same as in a unauthenticated and plain Diffie-Hellman key agreement. One message per participant is sufficient, which can be sent in any order and the message, the SIK key, is a single group element in $\mathbb{Z}_{\mathsf{N}}$. The computational cost is determined by the two exponentiations of $\mathsf{r}_{\mathsf{ID}_1}$ (and $\mathsf{r}_{\mathsf{ID}_2}$), in the online as well as in the offline case. A more detailed elaboration of the computational performance is done in Chapter 7.

3.4.2 With Multiple ID-PKGs

In the previous section, the four basic steps of the proposed key agreement scheme SSF were presented, where a single ID-PKG generates the public, shared parameters, and all identity keys. In this section, it will be shown how multi ID-PKG key agreement can be achieved with independent ID-PKGs.

Related Approaches. First of all, it is shortly resumed how the existing schemes handle multiple ID-PKGs. The identity-based key agreement scheme from McCullagh and Barreto [84] based on bilinear pairings is a good example for this purpose. In their approach, two distinct ID-PKGs, D_1 and D_2, must agree on a common elliptic curve E and two points on it, say P and Q. So, the ID-PKGs operate both on the same curve, which is a key point for the compatibility. Assume a third ID-PKG comes into play and wants to enable its users to communicate with the rest. Because the existing ID-PKGs most likely do not want to reset their systems and keys, the new ID-PKG is forced to overtake the used parameters $\mathsf{E}, \mathsf{P}, \mathsf{Q}$. Can the ID-PKG in this case be sure that these parameters were indeed chosen carefully? Even worse, the parameters can be chosen maliciously in the way that they contain a secret backdoor, that allows accessing the generated secret keys of the new ID-PKG. Consequently, this approach does not guarantee a fair setup for all ID-PKGs. Only if every involved ID-PKG can choose its parameters itself, they can be sure that the setup is sufficiently secure.

Proposed Approach. In the following, it is assumed without loss of generality that there are two domains D_1 and D_2. Their public parameters are $\mathrm{PSP}_1 = (\mathsf{N}_1, \mathsf{G}_1, \mathsf{R}_1, \mathsf{H}_1(\cdot))$ and $\mathrm{PSP}_2 = (\mathsf{N}_2, \mathsf{G}_2, \mathsf{R}_2, \mathsf{H}_2(\cdot))$, respectively. Every parameter can be chosen independently and even the case that either $(\mathsf{R}_2, \varphi(\mathsf{N}_1)) > 1$ or $(\mathsf{R}_1, \varphi(\mathsf{N}_2)) > 1$ holds is not critical, since no R-th roots must be computed regarding the other ID-PKGS's modulus. In the following, a participant $\mathsf{E}_{\mathsf{ID}_1}$ of D_1 wants to agree on a session key with a participant $\mathsf{E}_{\mathsf{ID}_2}$ of D_2. Therefore, $\mathsf{E}_{\mathsf{ID}_1}$ has to extend some of the involved integer to make them compatible with the PSP of domain D_2, i.e. PSP_2. In the proposed protocol, E_{ID} is able to perform the extension on its own without contacting a third party.

To reach compatibility with the foreign PSPs, each involved user executes an Extend algorithm, that extends its own identity key as well as the hash values of the remote identity. The Extend algorithm does the following:

The output of the multiple ID-PKG extension algorithm is an extended SIK ($\mathrm{eSIK}^{(1,2)}$) that can be used to perform a key agreement with a participant from domain D_2. (The notation $\mathrm{eSIK}^{(1,2)}$ represents an extended SIK between the domains D_1 and D_2.) The public shared parameters of the two involved participants are combined to the new set $\mathrm{PSP}_{1,2}$. This is done by multiplication of the first three items. However, this is not the only possibility. Gennaro et al. [48] suggested to use $\mathsf{R}_{1,2} = \mathrm{lcm}(\mathsf{R}_1, \mathsf{R}_2)$ and the

Algorithm 9 (SSF) $\mathsf{Extension}$ (view of $\mathsf{E}_{\mathsf{ID}_1}$)

```
Input: $PSP_1$, $PSP_2$, $\mathsf{d}_{\mathsf{ID}_1}$, $\mathsf{ID}_2$
Output: $\widetilde{\mathsf{d}}_{\mathsf{ID}_1}, \widetilde{\mathsf{H}_2}(\mathsf{ID}_2)$
1. $PSP_{1,2} := (\mathsf{N}_1 \cdot \mathsf{N}_2, \mathsf{G}_1 \cdot \mathsf{G}_2, \mathsf{R}_1 \cdot \mathsf{R}_2, \mathsf{H}_2)$
//Use the CRT to compute the integer $\widetilde{\mathsf{d}}_{\mathsf{ID}_1}$:
2a. $\widetilde{\mathsf{d}}_{\mathsf{ID}_1} \equiv \mathsf{d}_{\mathsf{ID}_1} \pmod{\mathsf{N}_1}$
2b. $\widetilde{\mathsf{d}}_{\mathsf{ID}_1} \equiv 1 \pmod{\mathsf{N}_2}$
//Use the CRT to compute the integer $\widetilde{\mathsf{H}_2}(\mathsf{ID}_2)$:
3a. $\widetilde{\mathsf{H}_2}(\mathsf{ID}_2) \equiv \mathsf{H}_2(\mathsf{ID}_2)^{\mathsf{R}_1} \pmod{\mathsf{N}_2}$
3b. $\widetilde{\mathsf{H}_2}(\mathsf{ID}_2) \equiv 1 \pmod{\mathsf{N}_1}$
return $eSIK^{(1,2)}_{\mathsf{ID}_1}$
```

Chinese Remainder result of

$$\mathsf{G}_{1,2} \equiv \begin{cases} \mathsf{G}_1 \pmod{\mathsf{N}_1} \\ \mathsf{G}_2 \pmod{\mathsf{N}_2} \end{cases}$$

However, using $\mathsf{G}_{1,2} = \mathsf{G}_1\mathsf{G}_2$ and $\mathsf{R}_{1,2} = \mathsf{R}_1\mathsf{R}_2$ does not lead to an insecure system, but only to slightly larger values. And if R_1 as well as R_2 is prime, these values are actually the same. However, when using $\mathsf{G}_{1,2}$ from the CRT, the proof of the protocol gets simplified, thus the reader should remember that in the proof section the CRT value rather than $\mathsf{G}_1 \cdot \mathsf{G}_2$ is used.

Algorithm 10 (SSF) $\mathsf{BuildSIK}_{\mathsf{MultIDPKG}}$

```
Input: $PSP_{1,2}$, $\widetilde{\mathsf{d}}_{\mathsf{ID}_1}$
Output: $eSIK^{(1,2)}_{\mathsf{ID}_1}$
1. $eSIK^{(1,2)}_{\mathsf{ID}_1} \equiv (\mathsf{G}_1 \cdot \mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_1}} \widetilde{\mathsf{d}}_{\mathsf{ID}_1} \pmod{\mathsf{N}_1\mathsf{N}_2}$
return $eSIK^{(1,2)}_{\mathsf{ID}_1}$
```

The key agreement is then performed by a similar algorithm as the standard $\mathsf{Compute}$ algorithm, but using the extended values and the combined set of PSPs.

It has to be shown that the key agreement algorithm between users from independent ID-PKGs is indeed correct.

Lemma 3.4.3 *The* SSF *key agreement algorithm between users from different ID-PKGs is correct.*

Proof 3.4.4 (of Lemma 3.4.3) *It has to be shown that the* $\mathsf{Compute}_{\mathsf{MultIDPKG}}$ *algorithm indeed outputs the same integer for both participants. More precisely, it has to*

Algorithm 11 (SSF) $\mathsf{Compute}_{\mathsf{MultIDPKG}}$ (view of $\mathsf{E}_{\mathsf{ID}_1}$)

```
Input for $E_{ID_1}$: $ID_2$, $PSP_{(1,2)}$, $eSIK^{(1,2)}_{ID_2}$, $r_{ID_1}$, $\widetilde{H_2}(ID_2)$
Input for $E_{ID_2}$: $ID_1$, $PSP_{(1,2)}$, $eSIK^{(1,2)}_{ID_1}$, $r_{ID_2}$, $\widetilde{H_1}(ID_1)$
Output: common integer $S$
1. $\left(((G_1 \cdot G_2)^{r_{ID_2}} \widetilde{d}_{ID_2})^{R_1 \cdot R_2} \widetilde{H_2}(ID_2)^{-1}\right)^{2r_{ID_1}} \equiv (G_1 \cdot G_2)^{2R_1R_2r_{ID_1}r_{ID_2}} \equiv S \pmod{N_1 \cdot N_2}$
2. $\left(((G_1 \cdot G_2)^{r_{ID_1}} \widetilde{d}_{ID_1})^{R_1 \cdot R_2} \widetilde{H_1}(ID_1)^{-1}\right)^{2r_{ID_2}} \equiv (G_1 \cdot G_2)^{2R_1R_2r_{ID_1}r_{ID_2}} \equiv S \pmod{N_1 \cdot N_2}$
return $S$
```

be shown that for modulo $(\mathsf{N}_1\mathsf{N}_2)$ *it holds:*

$$\left(((G_1 \cdot G_2)^{r_{ID_2}} \widetilde{d}_{ID_2})^{R_1 \cdot R_2} \widetilde{H_2}(ID_2)^{-1}\right)^{2r_{ID_1}} \equiv \left(((G_1 \cdot G_2)^{r_{ID_1}} \widetilde{d}_{ID_1})^{R_1 \cdot R_2} \widetilde{H_1}(ID_1)^{-1}\right)^{2r_{ID_2}} \equiv S$$

If the congruence above leaves the same remainders on both sides with respect to the moduli N_1 *and* N_2*, the CRT states that there will a unique integer* S *modulo* $(\mathsf{N}_1\mathsf{N}_2)$*, which is the output of the* $\mathsf{Compute}_{\mathsf{MultIDPKG}}$ *algorithm.*

For the left side, for modulo N_1*:*

$$\left(((G_1 \cdot G_2)^{r_{ID_2}} \widetilde{d}_{ID_2})^{R_1 \cdot R_2} \widetilde{H_2}(ID_2)^{-1}\right)^{2r_{ID_1}} \equiv \left((G_1 \cdot G_2)^{r_{ID_2}})^{R_1 \cdot R_2}\right)^{2r_{ID_1}} \equiv$$
$$(G_1 \cdot G_2)^{2R_1R_2r_{ID_1}r_{ID_2}} \equiv S_1$$

For the left side, for modulo N_2*:*

$$\left(((G_1 \cdot G_2)^{r_{ID_2}} \widetilde{d}_{ID_2})^{R_1 \cdot R_2} \widetilde{H_2}(ID_2)^{-1}\right)^{2r_{ID_1}} \equiv \left(((G_1 \cdot G_2)^{r_{ID_2}} \widetilde{d}_{ID_2})^{R_1 \cdot R_2} H_2(ID_2)^{-1}\right)^{2r_{ID_1}}$$
$$\equiv \left(((G_1 \cdot G_2)^{r_{ID_2} R_1 \cdot R_2} d_{ID_2}^{R_1 \cdot R_2}) H_2(ID_2)^{-1}\right)^{2r_{ID_1}}$$
$$\equiv \left(((G_1 \cdot G_2)^{r_{ID_2} R_1 \cdot R_2} H_2(ID_2)^{R_1}) H_2(ID_2)^{-1}\right)^{2r_{ID_1}}$$
$$\equiv \left((G_1 \cdot G_2)^{r_{ID_2} R_1 \cdot R_2}\right)^{2r_{ID_1}} \equiv \left((G_1 \cdot G_2)^{2r_{ID_1} r_{ID_2} R_1 \cdot R_2}\right) \equiv S_2$$

For the right side, for modulo N_1*:*

$$\left(((G_1 \cdot G_2)^{r_{ID_1}} \widetilde{d}_{ID_1})^{R_1 \cdot R_2} \widetilde{H_1}(ID_1)^{-1}\right)^{2r_{ID_2}} \equiv \left(((G_1 \cdot G_2)^{r_{ID_1}} \widetilde{d}_{ID_1})^{R_1 \cdot R_2} H_1(ID_1)^{-1}\right)^{2r_{ID_2}}$$
$$\equiv \left(((G_1 \cdot G_2)^{r_{ID_1} R_1 \cdot R_2} d_{ID_1}^{R_1 \cdot R_2}) H_1(ID_1)^{-1}\right)^{2r_{ID_2}}$$
$$\equiv \left(((G_1 \cdot G_2)^{r_{ID_1} R_1 \cdot R_2} H_1(ID_1)^{R_2}) H_1(ID_1)^{-1}\right)^{2r_{ID_2}}$$
$$\equiv \left((G_1 \cdot G_2)^{r_{ID_1} R_1 \cdot R_2}\right)^{2r_{ID_2}} \equiv \left((G_1 \cdot G_2)^{2r_{ID_1} r_{ID_2} R_1 \cdot R_2}\right) \equiv S_1$$

For the right side, for modulo N_2*:*

$$\left(((\mathsf{G}_1\cdot\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_1}}\widetilde{\mathsf{d}}_{\mathsf{ID}_1})^{\mathsf{R}_1\cdot\mathsf{R}_2}\widetilde{\mathsf{H}_1}(\mathsf{ID}_1)^{-1}\right)^{2\mathsf{r}_{\mathsf{ID}_2}}\ \left(((\mathsf{G}_1\cdot\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_1}})^{\mathsf{R}_1\cdot\mathsf{R}_2}\right)^{2\mathsf{r}_{\mathsf{ID}_2}} \equiv$$
$$\left((\mathsf{G}_1\cdot\mathsf{G}_2)^{2\mathsf{r}_{\mathsf{ID}_1}\mathsf{r}_{\mathsf{ID}_2}\mathsf{R}_1\cdot\mathsf{R}_2}\right)\equiv \mathsf{S}_2$$

Since the right and left sides leave the same remainders modulo N_1 *and* N_2*, respectively, a unique integer* S *can be obtained via the CRT.* □

Protocol Flow. Figure 3.2 illustrates the actual protocol flow. It is assumed that both participants possess the same set of public shared parameters. In the first step, $\mathsf{E}_{\mathsf{ID}1}$ computes its session initiation key using Algorithm 7. Subsequently, $\mathsf{E}_{\mathsf{ID}1}$ sends the generated SIK to $\mathsf{E}_{\mathsf{ID}2}$ describing the identity ID_1 as the packet's source. In the third step, $\mathsf{E}_{\mathsf{ID}2}$ generates a SIK on his part, which he sends to $\mathsf{E}_{\mathsf{ID}1}$ in step four appending his identity as the source.

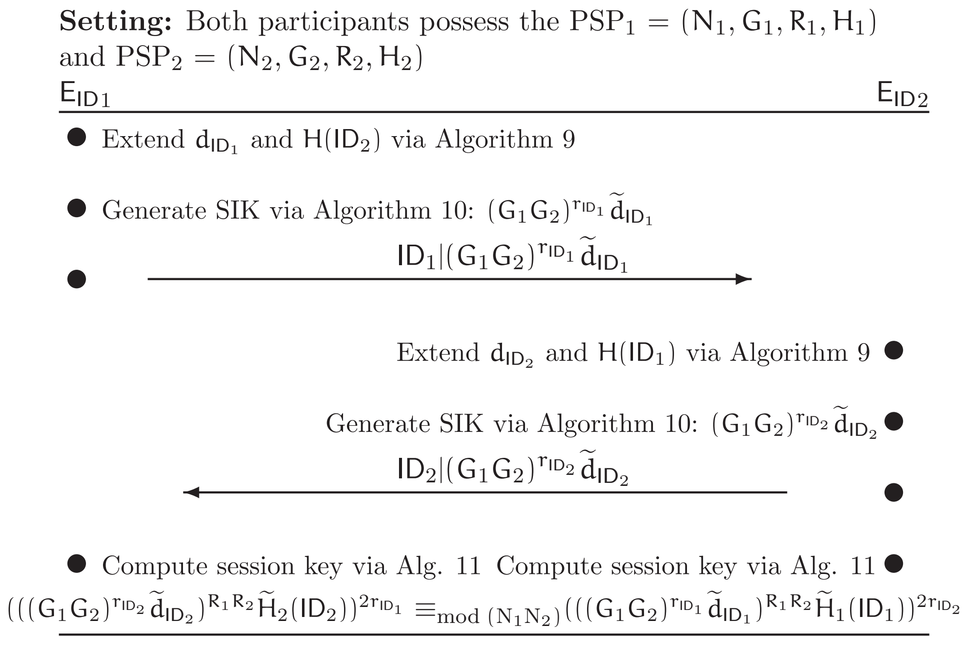

Setting: Both participants possess the PSP$_1$ = $(\mathsf{N}_1, \mathsf{G}_1, \mathsf{R}_1, \mathsf{H}_1)$ and PSP$_2$ = $(\mathsf{N}_2, \mathsf{G}_2, \mathsf{R}_2, \mathsf{H}_2)$
$\mathsf{E}_{\mathsf{ID}1}$
$\mathsf{E}_{\mathsf{ID}2}$
● Extend $\mathsf{d}_{\mathsf{ID}_1}$ and $\mathsf{H}(\mathsf{ID}_2)$ via Algorithm 9
● Generate SIK via Algorithm 10: $(\mathsf{G}_1\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_1}}\widetilde{\mathsf{d}}_{\mathsf{ID}_1}$
$\mathsf{ID}_1|(\mathsf{G}_1\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_1}}\widetilde{\mathsf{d}}_{\mathsf{ID}_1}$
Extend $\mathsf{d}_{\mathsf{ID}_2}$ and $\mathsf{H}(\mathsf{ID}_1)$ via Algorithm 9 ●
Generate SIK via Algorithm 10: $(\mathsf{G}_1\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_2}}\widetilde{\mathsf{d}}_{\mathsf{ID}_2}$ ●
$\mathsf{ID}_2|(\mathsf{G}_1\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_2}}\widetilde{\mathsf{d}}_{\mathsf{ID}_2}$
● Compute session key via Alg. 11
Compute session key via Alg. 11 ●
$(((\mathsf{G}_1\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_2}}\widetilde{\mathsf{d}}_{\mathsf{ID}_2})^{\mathsf{R}_1\mathsf{R}_2}\widetilde{\mathsf{H}}_2(\mathsf{ID}_2))^{2\mathsf{r}_{\mathsf{ID}_1}} \equiv_{\mathrm{mod}\ (\mathrm{N}_1\mathrm{N}_2)} (((\mathsf{G}_1\mathsf{G}_2)^{\mathsf{r}_{\mathsf{ID}_1}}\widetilde{\mathsf{d}}_{\mathsf{ID}_1})^{\mathsf{R}_1\mathsf{R}_2}\widetilde{\mathsf{H}}_1(\mathsf{ID}_1))^{2\mathsf{r}_{\mathsf{ID}_2}}$

Figure 3.2: The SSF protocol flow in the multi ID-PKG case.

Key Escrow. The term *key escrow* concerns the problem that another entity, e.g. a trusted authority, possesses the private key as well. This essentially cancels the meaning of a *private* key since it is not private anymore. When using IBC, the key generator is actually such a trusted authority that knows the user's private key. This is often mentioned as a killing argument by some potential adopters. However, in fact in almost all business environments that are based on public key infrastructures[2],

[2]This fact was learned during a talk with the PGP corporation, one of the world's leading companies

the private key of each employee is not generated by the employee itself, but by the company's administrators. This is necessary because the company needs to be able to decrypt data under unforeseen consequences, like for example, for the encrypted harddisk of a released employee. Thus, the application of IBC would not bring a new risk regarding key escrow.

Anyway, it would be a nice advantage if the ID-PKG would reduce some of its influence. There are some publications in which the ID-PKG forfeits the ability to decrypt the entire communication. For example, Goyal introduced a system [55] in which every identity is associated with an exponential number of identity keys. Via an *oblivious transfer protocol* (OTP), a user selects one of them. Based on the nature of an OTP, the ID-PKG does not learn which key the entity has chosen and a malicious ID-PKG can only select one of the exponential identity keys randomly. The argument is that if two different identity keys are identified in the system, it will be a proof that a malicious ID-PKG is running. Although this is a good idea, it does not solve the problem of key escrow. Furthermore, detecting two equal identity keys (= *private keys*) is questionable, since no one will present its private key without a very good reason.

Another approach is to use a threshold base generation process to compute the identity keys, as proposed by Boneh and Franklin [22]. This means, several ID-PKGs are used and no ID-PKG can generate an identity key on its own. Each one can only generate a part of the key and those different parts are combined by the entity to create the entire identity key. If the ID-PKGs cooperate among each other, the approach obviously fails. Hence, the ID-PKGs should be maintained by different authorities to prevent an easy way of cooperation. This approach is probably the best way, even if it circumvents the core problem. Furthermore, it adds the need for an additional infrastructure and initial cooperation between the ID-PKGs to agree on the parameters.

Note that in SSF, the power of the ID-PKG is reduced to active attacks. The ID-PKG cannot subvert the encryption if it only eavesdrops the communication, since the session key is based on the C-DHA. Other existing IBC schemes [84] state that they can setup their scheme in escrow mode as well as in no-escrow mode. However, this formulation is misleading. In their no-escrow mode, they only achieve the same security as in the proposed protocol. That means, it is secure except for active attacks, since their protocol in escrow-mode even allows the ID-PKG to passively decrypt the messages.

---

for selling PKI infrastructures.

3.5 Signatures

The continued proliferation of CPU, bandwidth and energy constrained devices increases the need for bit and CPU efficient cryptographic protocols. Multi-signatures allow multiple signers to jointly authenticate a message using a single compact signature. However, many applications require the public keys of the signers to be sent along together with the signature, partly defeating the effect of a compact signature [11]. While there are several multi-signature schemes, such as [91, 10] or [80], there are relatively few identity-based multi-signature schemes. In 2006, Gentry and Ramzan [51] and Chen et al. [33] presented identity-based multi-signatures (IBMSs) schemes, both based on pairings. In 2007, Bellare and Neven [11] showed how the Guillou-Quisqater scheme can be used as an IBMS. Bellare and Neven [11] have defined Bellare interactivity and non-interactivity as an important attribute of an IBMS. A non-interactive IBMS allows each participant to independently compute its share to the signature, and anyone can combine these shares into a compact signature. Interactive schemes require some form of cooperation between the entities, reducing the communication benefit of identity-based cryptography. The schemes of Chen et al. [33] and Bellare and Neven [11] require such an interaction. The scheme of Gentry and Ramzan [51] is non-interactive. None of the IBMSs are capable of using multiple ID-PKGs. This restricts their real world applicability, since in the mobile resource restricted scenarios where multi-signatures are particularly relevant, it is unlikely that a single trusted ID-PKG can be found. For instance, mobile phone operators are unlikely to agree on a single ID-PKG for all their customers due to conflicting business interests and management structures. To develop an IBMS for real world applications, the IBMS must be capable of working with multiple ID-PKGs with little or no cooperation between the operators of the ID-PKGs. Similar to Bellare and Neven's definition of interactivity for the signing entities, two classes of ID-PKGs for IBMS are defined. Dependent ID-PKGs must cooperate to function in unison, e.g. they must share secret parameters or share a single trusted root authority, whereas independent ID-PKGs can set up and operate without requiring shared secrets or trust between the ID-PKGs.

3.6 Related Protocols

Guillou-Quisquater/Bellare-Neven

The identity-based signature scheme of Guillou and Quisquater [60] has a similar setup as the proposed scheme. It was later extended to a multi-signature scheme (but with single ID-PKG) by Bellare and Neven [11]. The setup of the original scheme is shown next.

Algorithm 12 (QQ) Setup

```
Input: $k_1$,$k_2$
Output: {PSP, SP}
1. $P \xleftarrow{\$} \mathcal{PRIMES}(k_2)$
2. if $(P-1)/2$ is not prime goto 1
3. $Q \xleftarrow{\$} \mathcal{PRIMES}(k_2)$
4. if $(Q-1)/2$ is not prime goto 3
5. $N \leftarrow P \cdot Q$
6. $\nu \xleftarrow{\$} \{1, \min(P,Q)\}$
7. $H(\cdot) \leftarrow$ collision-resistant hash function, with $|H(\cdot)| < \nu$
8. return $\{(N,G,R,H),(P,Q)\}$
```

Public Shared Parameter of GQSS: PSP := (N, ν, H).
Secret Parameters of GQSS**:** SP := (P, Q).

The authors also use the IFP as the basic assumption for the protocol and keep the factorization as the ID-PKG's secret key. The secret identity key for each participant is extracted by $d_{ID} \equiv \mathrm{Red}(ID)^{-\nu^{-1}} \pmod{N}$. Note, that this key is similar to the identity key of Okamoto-Tanaka as well as the one of the proposed protocol, except for the additional inverse operation. Red$(\cdot)$ is here a reduction function, which can be instantiated by a hash function, for example. The generation of a signature is done via Algorithm 13 and the verifications steps are shown in Algorithm 14.

Algorithm 14 (QQ) SigVer

```
Input: PSP, $(d, z, m, ID)$
Output: true, if signature is valid, false otherwise
1. $\nu_l = d$
2. $\nu_r = H(z^{\nu}\,\mathrm{Red}(ID)^{d} \pmod{N}, m)$
3. return true if $\nu_l = \nu_r$ else return false
```

Algorithm 13 (QQ) SigGen

```
Input: PSP, $m$, $d_{ID}$
Output: $(d, z, m, ID)$
1. $r \xleftarrow{\$} \mathbb{Z}_N$
2. $t \equiv r^{\nu} \pmod N$
3. $d = H(t, m)$
4. $z \equiv r \cdot d_{ID}^{d} \pmod N$
5. return $(d, z, m, ID)$
```

The correctness can be easily verified, since

$$\begin{aligned} v_r &= H(z^{\nu}\,\mathrm{Red}(ID)^{d} \pmod N, m) = H(r^{\nu} d_{ID}^{\nu d}\,\mathrm{Red}(ID)^{d} \pmod N, m) \\ &= H(r^{\nu}\,\mathrm{Red}(ID)^{-\nu^{-1}\nu d}\,\mathrm{Red}(ID)^{d} \pmod N, m) = H(r^{\nu} \pmod N, m) \\ &= H(t, m) = d = v_l \end{aligned}$$

Extension from Bellare-Neven to Multi-Signatures. Bellare and Neven were the first authors who proved the security of the multi-signature extension of the Guillou-Quisquarter scheme. To accumulate several signatures on one document to one single signature S, they build the following product:

$$S \equiv \prod r_i^{\nu} \prod d_{ID_i}^{H(\prod r_i^{\nu} \pmod N), m)} \pmod N \tag{3.1}$$

However, this approach makes their multi-signature scheme clearly interactive, since the exponent $H(\prod r_i^{\nu} \pmod N), m)$, which must be known by all signers, can only be created if everyone knows all r_i.

3.7 SSF Signatures

In this section, it is demonstrated that a signature scheme can be build upon the already existing keys that also allows different participants to sign a single document.

3.7.1 Single Signatures

To be able to issue signatures with the SSF protocol, a small adaptation regarding the PSP has to be done. The adaptation concerns the integer R that must not be a prime number in this case. The new definition follows below:

Public, Shared Parameters (PSP). *The public, shared parameters of the signature scheme* SSF *is the quadruple* $\text{PSP} = (\mathsf{N}, \mathsf{G}, \mathsf{R}, \mathsf{H})$, $\mathsf{N} = \mathsf{PQ}$, $\mathsf{P}, \mathsf{Q} \in \mathbb{P}$. *The integer* N *is chosen to be a RSA integer.* G *is a generator of a large subgroup in* $\mathbb{Z}_\mathsf{N}$*.* R *is an integer with the property* $\gcd(\mathsf{R}, \varphi(\mathsf{N})) = 1$ *and* R *must have at least one factor* $\nu > 1$ *such that* R *can be written as* $\mathsf{R} = \nu\hat{\mathsf{R}}$ *with* $\hat{\mathsf{R}} > 1$*, with* $\nu, \hat{\mathsf{R}} \in \mathbb{N}$*. Finally,* H *is a collision free and secure hash function that maps the input to an element of* $\mathbb{Z}_\mathsf{N}$*.*

First, the basic version of the proposed signature scheme is presented. Afterwards, the scheme is extended in two steps. The first step enables multi-signatures and the second step enables multi-signatures in a setup of independent ID-PKGs. First, the PSP $(\mathsf{N}, \mathsf{G}, \mathsf{R}, \mathsf{H})$ are extended by another hash function $\hat{\mathsf{H}} : \{0,1\}^* \to \{0,1\}^w$, which produces a w-bit output. The bit length of the integer R is still k_1.

Next, the two basic algorithms are defined: $\mathsf{SigGen}((\mathsf{N}, \mathsf{G}, \mathsf{R}, \mathsf{H}), m, \mathsf{k}, \mathsf{d}_{\mathsf{ID}}, \mathsf{ID}) \to \mathcal{S}$ and $\mathsf{SigVer}((\mathsf{G}, \mathsf{N}, \mathsf{R}, \mathsf{H}), \mathcal{S}) \to \{\mathsf{true}|\mathsf{false}\}$. The SigGen-Algorithm (shown in Algorithm 15) uses the following input parameters:

1. PSP: public shared parameters
2. m: the message to be signed
3. k: security parameter
4. d_{ID}: private identity key of the executing entity
5. ID: the identity of the executing entity

The algorithm returns the signature $\mathcal{S}$ of the message m signed by the identity ID.

The SigVer-Algorithm shown in Algorithm 16 uses the following input parameters:

1. PSP: public shared parameters
2. $\mathcal{S}$: signature to be verified

It returns true if the signature is valid and false otherwise.

First, it has to be shown that the verification algorithm is correct. This means, if all

Algorithm 15 (SSF) SigGen

```
Input: PSP, $m$, $k$, $d_{ID}$, $ID$
Output: $(s_1, s_2, m, ID) = \mathcal{S}$
1. $h \leftarrow H(m)$
2. check if $R \nmid h$, if yes return failure
//$k$ is a security parameter that determines the maximum bit length of $\alpha$
3. $\alpha \overset{\$}{\leftarrow} \{1, ..., 2^k\}$
4. Compute $G^{\alpha} \pmod N$
5. $s_1 \equiv G^{h\alpha} d_{ID} \pmod N$
6. $s_2 \equiv G^{\hat{R}\alpha} \pmod N$
7. return $(s_1, s_2, m, ID)$
```

Algorithm 16 (SSF) SigVer

```
Input: PSP, $(s_1, s_2, m, ID)$
Output: true, if signature is valid, false otherwise
1. $h \leftarrow H(m)$
2. $v_l \equiv s_1^R H(ID)^{-1} \pmod N$
3. $v_r \equiv s_2^{vh} \pmod N$
4. return true if $v_l = v_r$
```

operations are done as specified and no tampering occurred during transmissions, the algorithm returns true. Thus, it has to be shown that $v_l = v_r$:

Lemma 3.7.1 (Correctness) *The SSF Signature Verification algorithm is correct.*

Proof 3.7.2 *For v_l and v_r as defined in Algorithm 16 it holds* $\pmod N$*:*

$$v_l \equiv s_1^R H(ID)^{-1} \equiv \left(G^{h\alpha} d_{ID}\right)^R H(ID)^{-1} \equiv G^{Rh\alpha} H(ID) H(ID)^{-1} \equiv G^{Rh\alpha} \equiv s_2^h \equiv v_r$$

3.7.2 Multi-Signatures

If an identity-based signature scheme enables several identities to contribute to a signature, it is called an *identity-based multi-signature* scheme. Obviously, any signature scheme could be converted to an IBMS by concatenating several signature to one multi-signature. For efficient IBMS schemes, the generated n-identity signature (a signature where n identities have contributed) should be less in size than n times a 1-identity signature. Furthermore, multi-signature scheme can be also distinguished in the way they allow each signer to contribute. If a scheme allows independent signatures in a way that the signers do not have to interact to create signature, the scheme is called *non-interactive*. The property of non-interactive is important in scenarios where the channel between the signers is only one-way. For example, multi-level security architectures only allow users from different access levels to inter one-way. Interactive multi-signature would reduce its operational area only to one layer instead of the whole system. The IBMS presented below is both independent and non-interactive and thus is the first IBMS that can fully benefit from the advantages of multi-signatures and is applicable to real world problems.

It will be shown that the signature scheme allows n signers to sign a message m in any order and that the i-th signer is not forced to verify the signature received so far, but can independently contribute to the signature, thus earns the property of being non-interactive. Even the verifier does not need to know the order the multi-signature has been created.

We write d_{ID_i} with $1 \leqslant i \leqslant n$ for the participating identities. All corresponding entities $E_{\mathsf{ID}i}$ (the signers) share the same set of PSP and know the message m. We write

$$\mathcal{S}^{(1,2,\ldots,i-1)} = (s_1^{(1,2,\ldots,i-1)}, s_2^{(1,2,\ldots,i-1)}, m, \{\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_{i-1}\})$$

for the $(i-1)$-th signature. The associated generation algorithm is an extension of the basic generation algorithm (Algorithm 15) using the $(i-1)$-th signature as an additional input parameter.

Algorithm 17 (SSF) MultSigGen (from the view of the i-th signer)

```
Input: PSP, $k$, $d_{\mathsf{ID}}$, $\mathsf{ID}$, $\mathcal{S}^{(1,2,\ldots,i-1)}$
Output: $(s_1^{(1,2,\ldots,i)}, s_2^{(1,2,\ldots,i)}, m, \{\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_i\}) = \mathcal{S}^{(1,2,\ldots,i)}$
1. $h \leftarrow H(m)$
2. check if $R \nmid h$, if yes return failure
3. $\alpha_i \overset{\$}{\leftarrow} \{1, \ldots, 2^k\}$
4. Compute $G^{\alpha_i} \pmod N$
5. $s_1^{(1,2,\ldots,i)} \equiv s_1^{(1,2,\ldots,i-1)} G^{h\alpha_i} d_{\mathsf{ID}_i} \pmod N$
6. $s_2^{(1,2,\ldots,i)} \equiv s_2^{(1,2,\ldots,i-1)} G^{\frac{R}{v}\alpha_i} \pmod N$
7. return $(s_1^{(1,2,\ldots,i)}, s_2^{(1,2,\ldots,i)}, m, \{\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_i\})$
```

The multi-signature algorithm differs from the basic algorithm in the 5-th and 6-th line. Here, there are two additional factors, which come from the $(i-1)$-th signature. The multi-signature algorithm encompasses the single signature algorithm through these factors. The single signature case can be replicated by setting $s_1^{(1,2,\ldots,i-1)} = 1 = s_2^{(1,2,\ldots,i-1)}$. The verification algorithm for the multi-signature case is shown in Algorithm 18.

Algorithm 18 (SSF) MultSigVer

```
Input: PSP, $\mathcal{S}^{(1,2,...,n)} = (s_1^{(1,2,...,n)}, s_2^{(1,2,...,n)}, m, \{ID_1, ID_2, ..., ID_n\})$
Output: true, if signature is valid, false otherwise
1. $h \leftarrow H(m)$
2. $v_l \equiv (s_1^{(1,2,...,n)})^R \prod_{j=1}^{n} H(ID_j)^{-1} \pmod{N}$
3. $v_r \equiv (s_2^{(1,2,...,n)})^{vh} \pmod{N}$
4. return true if $v_l = v_r$ else return false
```

The difference of the multi-signature verification to the basic case is in Line 2, where the verifier of a n-identity signature has to build the product over all contributed identities. It can be seen that the order of creation does not need to be known because of the involved commutative operations.

Signature Size. The size of a n-identity signature is a measure for its efficiency. The size of a $(i-1)$-th signature and a i-th signature differs by the length of the identity string ID_i, assuming that the two group elements roughly have the same size according to their random characteristic.

$$[\text{Successive Signature Difference}] \quad |\mathcal{S}^{(1,2,...,i)}| - |\mathcal{S}^{(1,2,...,i-1)}| \approx |ID_i|$$

Thus, the total size of a n-identity signature is

$$[\text{Average Signature Size (bit)}] \quad |\mathcal{S}^{(1,2,...,n)}| \approx 2\frac{\log_2 N}{2} + \log_2 m + \sum_{j=1}^{n} |ID_j|$$

covering the 2 group elements (being both roughly of length $|N|/2$), the message m and all involved identities.

Fast Signature Aggregation. In the way the scheme is constructed, it allows to combine the partial signatures in any order and subparts can be put to together by any unit. If the message to be signed is initially known, then fast structures for signature aggregation can be created. Figure 3.3 shows a simple binary tree approach. If in each layer of the tree all participants are acting simultaneously, a significant speedup can be achieved.

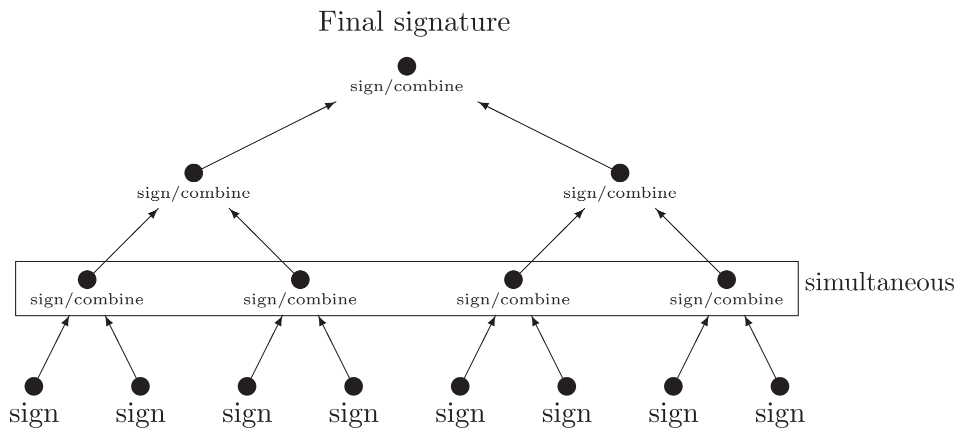


Figure 3.3: Fast signature aggregation via binary structure.

An application would be, for example, Dynamic Hash Tables networks, where all replication repositories are signing their content. In this way, they can create a multi-signature in fast way to prove that they all acknowledge to correct status of the stored content.

3.7.3 Multi-Signatures with Multiple ID-PKGs

In this section, the multi-signature algorithm is further extended to allow the use of multiple independent ID-PKGs. This is a critical issue for identity-based cryptography in general, since the sensitive nature of the ID-PKG makes it highly unlikely that competing organizations will cooperate in the setup and operation of their ID-PKGs. However, customers from multiple organizations need to be able to sign a document together. In general IBC systems there are efforts underway to mitigate the problem of multiple ID-PKGs, however, apart from [108] they all require the ID-PKGs to trust either each other and use shared secrets or a trust hierarchy with a single trusted root entity [16, 24, 52, 64, 84].

In the following, the first IBMS that allows independent (i.e. no shared secret, no inter ID-PKG trust and no trust hierarchy) operation of multiple ID-PKGs is presented. It has the desired attributes of compactness, non-interactivity and independence of the ID-PKGs. It does not require interaction between the signers, and the message can be signed in any order and the final signature can be constructed by any entity. It also allows singing entities to have received their identity keys from multiple different ID-PKGs and it does not require the ID-PKGs to cooperate, i.e. share secrets or a single trusted authority. To speed up the operation of the IBMS, precomputation techniques [54] are used for all exponentiation steps. A security analysis is presented using the random oracle model. It will be shown that forging a signature is not possible assuming the validity of the RSA assumption.

Each ID-PKG can create its private parameters and create identity keys independently of any other ID-PKG. For reasons of simplicity it is required that the non-critical public parameter R and the hash-function H are the same for all PSP. Requiring these two parameters to be the same for all ID-PKGs does not affect the security of the ID-PKGs and does not require the ID-PKGs to share any secret knowledge or form any kind of trust relationship. If for any reason different Rs or Hs are required, this is also possible. The protocol then loses the non-interactivity feature for signing entities.

In the multiple-ID-PKG scenario, participants possess identity keys created by multiple ID-PKGs using different secret parameters tied to different PSPs. To deal with the multiple PSPs, an adaptation of Algorithm 9 is presented which extends the identity key to be valid for multiple PSPs. This extension algorithm can be executed by each entity independently. This Extension algorithm is shown in Algorithm 19.

Algorithm 19 (SSF) IdKeyExt (from the view of the i-th signer)

```
Input: $PSP_1, PSP_2, ..., PSP_w, d_{ID_i}$
Output: $\widetilde{d}_{ID_i}$
//$N_j$ is part of $PSP_j$
1. Use the Chinese Remainder Theorem to calculate the integer $\widetilde{d}_{ID_i}$
   by solving the system of $w$ simultaneous congruences:
2. $\widetilde{d}_{ID_i} \equiv d_{ID_i} \pmod{N_i}$
3. $\widetilde{d}_{ID_i} \equiv 1 \pmod{N_j}, \forall j \neq i$
4. return $\widetilde{d}_{ID_i}$
```

Algorithm 20 (SSF) HashValExt (from the view of the i-th signer)

```
Input: $PSP_1, PSP_2, ..., PSP_w, ID_i$
Output: $\widetilde{H}(ID_i)$
//$N_j$ is part of $PSP_j$
1. Use the Chinese Remainder Theorem to calculate the integer $\widetilde{H}(ID_i)$
   by solving the system of $w$ simultaneous congruences:
2. $\widetilde{H}(ID_i) \equiv H(ID_i) \pmod{N_i}$
3. $\widetilde{H}(ID_i) \equiv 1 \pmod{N_j}, \forall j \neq i$
4. return $\widetilde{H}(ID_i)$
```

The HashValExt algorithm is the counterpart to the IdKeyExt algorithm and ensures that the hash values from the identity string fit to the extended identity keys that are involved in the signature. Both extension algorithms make use of the Chinese Remainder Theorem. It can be assumed that all involved N_i's are co-prime, thus the CRT can compute the unique integers $\widetilde{d}_{ID_i}$ and $\widetilde{H}(ID_i)$ efficiently. Note that each signer can contribute independently to the signature, but the signer must know the message m and the involved PSPs.

Finally, the generation algorithm for a multi-signature with multiple ID-PKGs is shown in Algorithm 21 and demonstrates the actions of the i-th signer. Similar to the algorithms for a single ID-PKG, the signer now has to use its extended identity key, the product of all generators G_i and the product of all N_i.

Algorithm 21 (SSF) $\mathsf{MultSigGen}_{\mathsf{MultIDPKG}}$ (from the view of the i-th signer)

```
Input: $\mathcal{S}^{(1,2,\dots,i-1)}$, PSP$_1$, PSP$_2$,...,PSP$_w$, $k$, $d_{ID_i}$, $ID$
Output: $\mathcal{S}^{(1,2,\dots,i)}$
1. $\widetilde{d}_{ID_i} \leftarrow \mathsf{IdKeyExt}(PSP_1, PSP_2, \dots, PSP_w, d_{ID_i})$
2. $h \leftarrow H(m)$
3. check if $R \nmid h$, if yes return failure
4. $\alpha_i \xleftarrow{\$} \{1, \dots, 2^k\}$
5. $s_1^{(1,2,\dots,i)} = s_1^{(1,2,\dots,i-1)} \left(\prod_{j=1}^{w} G_j\right)^{h\alpha_i} \widetilde{d}_{ID_i} \pmod{\prod_{j=1}^{w} N_j}$
6. $s_2^{(1,2,\dots,i)} = s_2^{(1,2,\dots,i-1)} \left(\prod_{j=1}^{w} G_j\right)^{\frac{R}{v}\alpha_i} \pmod{\prod_{j=1}^{w} N_j}$
7. return $\mathcal{S}^{(1,2,\dots,i)} = (s_1^{(1,2,\dots,i)}, s_2^{(1,2,\dots,i)}, m, \{ID_1, \dots, ID_i\})$
```

The verification algorithm, shown in Algorithm 22, is similar to the previous versions. The verifier uses the extended hash values as well as the product of all N_i for the modulus.

Algorithm 22 (SSF) $\mathsf{MultSigVer}_{\mathsf{MultIDPKG}}$

```
Input: $\mathcal{S}^{(1,2,\dots,n)}$, PSP$_1$, PSP$_2$,...,PSP$_w$, $\{ID_1, \dots, ID_n\}$
Output: $\widetilde{d}_{ID_i}$
1. For each $ID_i$ do $\widetilde{H}(ID_i) \leftarrow \mathsf{HashValExt}(PSP_1, PSP_2, \dots, PSP_w, ID_i)$
2. $v_l \equiv (s_1^{(1,2,\dots,n)})^R \prod_{j=1}^{n} \widetilde{H}(ID_j)^{-1} \pmod{\prod_{j=1}^{w} N_j}$
3. $v_r \equiv (s_2^{(1,2,\dots,n)})^{vh} \pmod{\prod_{j=1}^{w} N_j}$
4. return true if $v_l = v_r$ else return false
```

Before presenting the proof of the proposed IBMS schemes, it will be shown that the multi-signature algorithm in the multiple ID-PKG case is correct.

Lemma 3.7.3 (Correctness) *The* $\mathsf{MultSigVer}_{\mathsf{MultIDPKG}}$ *is correct.*

Proof 3.7.4 *Verification follows Algorithm 22, thus the verifier computes:*

$$v_l = (s_1^{(1,2,\dots,n)})^R \prod_{j=1}^n \widetilde{H}(ID_j)^{-1} \equiv$$
$$\left(\prod_{j=1}^w G_j\right)^{Rh(\sum_{j=1}^n \alpha_j)} \prod_{j=1}^n \widetilde{d}_{ID_j}^R \prod_{j=1}^n \widetilde{H}(ID_j)^{-1} \pmod{\prod_{j=1}^w N_j}$$

and

$$v_r = (s_2^{(1,2,\dots,n)})^{vh} = \left(\prod_{j=1}^w G_j\right)^{Rh(\sum_{j=1}^n \alpha_j)} \pmod{\prod_{j=1}^w N_j}$$

It has to be shown that $\prod_{j=1}^n(\widetilde{d}_{ID_j})^R \prod_{j=1}^n \widetilde{H}(ID_j)^{-1} \equiv 1 \pmod{\prod_{j=1}^w N_j}$. *Since for* $i \neq j$ *it holds*

$$\widetilde{d}_{ID_j} \equiv \widetilde{H}(ID_j) \equiv 1 \pmod{N_i}$$

and for $i = j$

$$(\widetilde{d}_{ID_i})^R \equiv H(ID_i) \equiv \widetilde{H}(ID_i) \pmod{N_i}$$

the CRT guarantees to find a unique number, which proves the correctness.

3.8 Summary

In this chapter, the SSF scheme was presented. The basic scheme is actually comparable to the OkTa scheme, but the flaws in the construction were removed. Then, the scheme was extended to handle multiple ID-PKGs that can act independently of each other. Also, no participant is forced to get another set of keys in order to communicate with an entity of a foreign ID-PKG. The Chinese Remainder Theorem was used to transfer the identity-key d_{ID}, which is an element in $\mathbb{Z}_{N_1}$, to a unique element in $\widetilde{d}_{ID} \in \mathbb{Z}_{N_1 N_2}$ by adding the requirement $\widetilde{d}_{ID} \equiv 1 \pmod{N_2}$.

In the second part, a signature scheme based on the already existing SSF keys was introduced. Therefore, the PSP of the original key agreement case were modified slightly. Note, that these modified PSP can be also used in the key agreement case. The adaptation is that the public exponent R is **not** allowed to be a prime. This is necessary to show that in any case the proof of the scheme can be reduced to the RSA assumption (see next chapter). The basic signature scheme was then enhanced to issue multi-signatures, which essentially is a successive execution of the basic scheme by different signers. Finally, the novel multiple ID-PKG treatment to allow multi-signatures, each from multiple ID-PKGs, was applied to the signature scheme.

4 Security Analysis

"For those who believe, no proof is necessary.
For those who don't believe, no proof is possible."
Stuart Chase

4.1 Introduction

After having presented the entire SSF scheme, it is time to show that the scheme is actually secure regarding standard definitions from the literature. First, we show that the basic key agreement algorithms of the SSF scheme form a secure and authenticated key agreement protocol. The proof reduces the security of the scheme to the RSAA, the C-DHA and the GAP-DHA. This means that an attacker who is capable of breaking the key agreement in polynomial time can also break one of these NP-hard assumptions in polynomial time. Second, the security of the key agreement with multiple ID-PKGs is proven in a similar way, by utilizing the arguments of Gennaro et al. [48].

The later part of the chapter contains all the proofs regarding the SSF signature scheme. In this case, the security is reduced to the RSAA. Three proofs are presented: one for the single signature / single ID-PKG case, one for the multi-signature / single ID-PKG case and one for the multi-signature / multiple ID-PKG case.

The material presented in this chapter has been published in [110] and [109].

4.2 SSF with Single ID-PKG

In this section, it will be proven that SSF is indeed a secure and authenticated key agreement protocol. For this purpose, the Canetti-Krawczyk Model (CKM) as described in Chapter 2, Section 2.3.3, Definition 2.3.5 is used.

Theorem 4.2.1 (SSF **- single ID-PKG)** *Based on the* RSAA*, the* C-DHA *and the* GAP-DHA *assumptions,* SSF *is a secure and authenticated key agreement protocol in the sense of the CKM, if the hash function* H *is modeled as a programmable, random oracle.*

Proof 4.2.2 *The proof is done via reduction: It is shown that an adversary $\mathcal{A}$ who is able to break the* SSF *protocol with non-negligible probability, can be utilized by a simulator* Sim *to break the* C-DHA *and the* GAP-DHA *with non-negligible probability. The goal of the adversary is always to get the session key* S*, whereas* Sim *simulates the* SSF *protocol against the adversary $\mathcal{A}$.*

We distinguish between two cases: First, a passive adversary is assumed. In this case, we show that Sim *is able to break the* C-DHA*. In the second case, we assume an active adversary who can manipulate packets in any way. In this case, we show that we can build an algorithm that can break the* RSAA *with non-negligible probability.*

Case 1: A Matching Test Session. *Since* Sim *simulates the whole* SSF *world, it is also responsible to generate the private keys of each user. Although* Sim *does not know the ID-PKG's secret (*P *and* Q*), he can simulate the private keys by $\rho_{ID}^{R} \pmod{N} := H(ID)$, where ρ_{ID} is a random integer that is chosen differently for each* ID*. By this construction,* Sim *knows the private identity key of each user, which is $\rho_{ID} = d_{ID}$. Furthermore,* Sim *changes the generator* G *to $\hat{G} \equiv G^{R} \pmod{N}$. We now assume* Sim *is facing the problem to compute G^{uv} from G^{u} and G^{v} (that is an instance of the* C-DHA*) and utilizes the* SSF*-breaking adversary $\mathcal{A}$ for this purpose.* Sim *is prepared to run* m *sessions in total, for some polynomially bounded integer* m*. If the attacker does not decide to attack the protocol during these runs,* Sim *resets the system and restarts.*

After starting the protocol, Sim *initiates communications between various participants and answers all* ***State Reveal***, ***Session Key*** *and* ***Party Corruption*** *queries honestly to the attacker.*

Sim *guesses that in the* i*-th protocol execution ($i < m$), $\mathcal{A}$ will eavesdrop the communication and uses the gained information to guess the session key and breaks the* SSF *protocol with success probability a_{succ}.*

We assume that this i*-th session takes place between Alice and Bob, with Alice being* *the initiator.* Sim *constructs the SIK in this round via*

$$\mathrm{SIK}_{\mathsf{Alice}} \equiv \hat{\mathsf{G}}^{\mathsf{u}/\mathsf{R}} \mathsf{d}_{\mathsf{Alice}} \pmod{\mathsf{N}} \tag{4.1}$$

and

$$\mathrm{SIK}_{\mathsf{Bob}} \equiv \hat{\mathsf{G}}^{\mathsf{v}/\mathsf{R}} \mathsf{d}_{\mathsf{Bob}} \pmod{\mathsf{N}} \tag{4.2}$$

Since $\mathcal{A}$ is assumed to be passive, Bob and Alice have a matching conversation and successfully agreed onto a session key, which is according to the Compute *algorithm*

$$\mathsf{H}(\mathsf{S}) = \mathsf{H}((\mathrm{SIK}^{\mathsf{R}}_{\mathsf{Alice}} \mathsf{H}(\mathsf{Alice})^{-1})^{\mathsf{v}/\mathsf{R}}). \tag{4.3}$$

Sim *does not know this value, but he knows that it must look like this. Further substitution shows that this is actual equivalent to*

$$\mathsf{H}(\hat{\mathsf{G}}^{\mathsf{R}\mathsf{u}/\mathsf{R}} \rho^{\mathsf{R}}_{\mathsf{Alice}} \mathsf{H}(\mathsf{Alice})^{-1})^{\mathsf{v}/\mathsf{R}} \equiv \mathsf{H}(\hat{\mathsf{G}}^{\mathsf{uv}/\mathsf{R}}) \equiv \mathsf{H}(\mathsf{G}^{\mathsf{uv}}). \tag{4.4}$$

The session key between Alice and Bob is now the hash value of G^{uv}*, but which is still unknown to* Sim*. However, since $\mathcal{A}$ needs to ask the random oracle about the hash value* $\mathsf{H}(\mathsf{G}^{\mathsf{uv}})$ *to distinguish the result from random,* Sim *learns this value, too. Thus,* Sim *breaks the* C-DHA *since he found* G^{uv} *from* G^{u} *and* G^{v} *within the polynomially bounded number of queries of $\mathcal{A}$. Thus, the probability for* Sim *to break the* C-DHA *is*

$$\Pr[\mathsf{Sim}(\mathsf{G}^{\mathsf{u}}, \mathsf{G}^{\mathsf{v}}, \mathsf{N}) = \mathsf{G}^{\mathsf{uv}}] = \frac{\mathsf{a}_{\mathsf{succ}}}{\mathsf{m}} \tag{4.5}$$

which is non-negligible if $\mathsf{a}_{\mathsf{succ}}$ *is non-negligible.*

Case 2: No Matching Test Session. *In this case, no matching session between Alice and Bob can be assumed, since the attacker $\mathcal{A}$ could have manipulated all messages, which probably leads to different keys at Alice and Bob. However, we show that an attacker who breaks the* SSF *protocol in this case successfully, can be utilized to forge RSA signatures. Here, a forger $\mathcal{F}$, adopts the role of the simulator. He behaves exactly like the simulator* Sim *above and knows the private keys for all participants except the one of Bob. The goal of $\mathcal{F}$ is to compute the private key for Bob without knowing the factorization of* N.

By knowing all other private keys, the forger can construct all SIKs during a key agreement, except when he simulates key agreements between Bob and another participant. However, whenever Bob interacts with another participant, say Charly, $\mathcal{F}$ must be able to answer session key queries. Even $\mathcal{F}$ is not in possession of Bob's private key, he can *compute the session key between Bob and Charly by using Charly's private key only. To see this, we can assume that Bob's SIK is*

$$\text{SIK}_{\mathsf{Bob}} \equiv \mathsf{G}^{\mathsf{y}} \mathsf{d}_{\mathsf{Bob}} \pmod{\mathsf{N}} \tag{4.6}$$

where Charly's SIK can be written as

$$\text{SIK}_{\mathsf{Charly}} \equiv \mathsf{G}^{z} \mathsf{d}_{\mathsf{Charly}} \pmod{\mathsf{N}} \tag{4.7}$$

with y *and* $\mathsf{d}_{\mathsf{Bob}}$ *unknown to* $\mathcal{F}$[1]. $\mathcal{F}$ *can compute* $(\mathsf{SIK}^{\mathsf{R}}_{\mathsf{Bob}} \mathsf{H}(\mathsf{Bob})^{-1}) \equiv \mathsf{G}^{\mathsf{R}\mathsf{y}}$. *Since* $\mathcal{F}$ *knows* G^{z}, *he can respond always with the correct session key value* $\mathsf{G}^{2\mathsf{R}\mathsf{y}z}$.

More problematic are the session keys at Bob, which were generated by the attacker pretending to come from Charly, since $\mathcal{F}$ *does not know Bob's private key nor the secret exponent* z *in this case. Moreover, the session key from Charly (i.e. from* $\mathcal{A}$*) is not to be guaranteed to be an element of the group generated by* G*. However, since* $\mathsf{N} = \mathsf{PQ} = (2\mathsf{P}' + 1)(2\mathsf{Q}' + 1)$, *with* $\mathsf{P}, \mathsf{P}', \mathsf{Q}, \mathsf{Q}' \in \mathbb{P}$, *the malicious SIK generated by* $\mathcal{A}$ *can be written as*

$$\text{SIK}_{\mathsf{Charly}} \equiv \delta \mathsf{G}^{z} \mathsf{d}_{\mathsf{Charly}} \pmod{\mathsf{N}},$$

with δ *an element of order* 2 *in* $\mathbb{Z}_{\mathsf{N}}$. *We need to show how* $\mathcal{F}$ *will respond to a session key query by* $\mathcal{A}$: $\mathcal{F}$ *uses the knowledge of Charly's private key to compute* $\mathsf{G}^{2z} \equiv \gamma^{2}/\mathsf{d}^{2}_{\mathsf{Chary}}$ *and* $\mathsf{G}^{\mathsf{R}\mathsf{y}}$ *as already shown above. He then checks if one of the past queries* Q *of the attacker to the oracle satisfies* $\mathsf{DH}(\mathsf{G}^{2z}, \mathsf{G}^{\mathsf{R}\mathsf{y}}) = \mathsf{Q}$. *If so,* $\mathcal{F}$ *answers to a session key query with* $\mathsf{H}(\mathsf{Q})$, *otherwise he answers with a random integer.*

Simulation of the i***-th run.*** *In this run, the attacker tries to break the* SSF *protocol. Assume that the run takes place between Alice and Bob, with Bob being the initiator.* $\mathcal{A}$ *intercepts the packets and manipulates them in an arbitrary way. Since the attacker succeeds in this run, he outputs the correct session key, which is* $\mathsf{G}^{2\mathsf{R}\mathsf{x}\mathsf{y}}$.

Next, we show how this knowledge can be used to compute Bob's secret key, that means extracting the R*-th root out of* $\mathsf{H}(\mathsf{Bob})$. *Therefore,* $\mathcal{F}$ *makes the session initiation key from Alice as well as the base* G *dependent on Bob's identity.*

$$\mathsf{G} \equiv (\mathsf{rH}(\mathsf{Bob}))^{2\mathsf{R}}, \mathsf{H}(\mathsf{Alice}) \equiv \mathsf{s}^{\mathsf{R}}, \mathsf{SIK}_{\mathsf{Alice}} \equiv (\mathsf{rH}(\mathsf{Bob}))^{\mathsf{f}} \mathsf{s}$$

with r, s *being random elements in the group of* G, *as well as* f *being a random integer*

[1]Note, because $\mathsf{H}(\mathsf{Bob})$ maps into the group generated by G, such an y is guaranteed to exists.

co-prime to R. *Note, using this we can also write* $\mathsf{SIK}_{\mathtt{Alice}} = \mathsf{G}^{2^{-1}\mathsf{f}/\mathsf{R}}\mathsf{H}(\mathsf{Bob})^{1/\mathsf{R}} \equiv \mathsf{G}^{2\mathsf{x}}\mathsf{H}(\mathsf{Bob})^{1/\mathsf{R}}$. *If the attacker sends an arbitrary session initiation key* $\mathsf{SIK}_{\mathtt{Bob}}$, *the session key for this session is equal to* $\mathsf{K} = (\mathsf{SIK}^{\mathsf{R}}_{\mathtt{Bob}} \cdot \mathsf{H}(\mathsf{Bob})^{-1})^{2\mathsf{x}} \equiv (\mathsf{SIK}^{\mathsf{R}}_{\mathtt{Bob}} \cdot \mathsf{H}(\mathsf{Bob})^{-1})^{\mathsf{f}/\mathsf{R}}$, *thus*

$$\mathsf{K}^{\mathsf{R}} \equiv (\mathsf{SIK}^{\mathsf{R}}_{\mathtt{Bob}} \cdot \mathsf{H}(\mathsf{Bob})^{-1})^{\mathsf{f}} \Leftrightarrow \mathsf{d}^{\mathsf{f}}_{\mathtt{Bob}} \equiv \mathsf{SIK}^{\mathsf{f}}_{\mathtt{Bob}}\mathsf{K}^{-1}$$

$\mathcal{F}$ learns this session key from $\mathcal{A}$'s queries to the random oracle. With the help of K, *$\mathcal{F}$ knows two different powers of Bob's private key: He knows both,* $\mathsf{d}^{\mathsf{R}}_{\mathtt{Bob}} \equiv \mathsf{H}(\mathsf{Bob})$ *and* $\mathsf{d}^{\mathsf{f}}_{\mathtt{Bob}} \equiv \mathsf{SIK}^{\mathsf{f}}_{\mathtt{Bob}}\mathsf{K}^{-1}$. *Since* R *and* f *are co-prime, $\mathcal{F}$ can compute* $a\mathsf{R} + b\mathsf{f} = 1$ *using the Extended Euclidean Algorithm. Afterwards, he reveals* $\mathsf{d}_{\mathtt{Bob}}$ *by*

$$(\mathsf{d}^{\mathsf{R}}_{\mathtt{Bob}})^{a} \cdot (\mathsf{d}^{\mathsf{f}}_{\mathtt{Bob}})^{b} \equiv \mathsf{H}(\mathsf{Bob})^{a} \cdot (\mathsf{SIK}^{\mathsf{f}}_{\mathtt{Bob}}\mathsf{K}^{-1})^{b} \equiv \mathsf{d}_{\mathtt{Bob}}$$

The success probability for $\mathcal{F}$ is the same as the probability for Sim *in the first case, since the success only depends on the guessed round and the winning probability of $\mathcal{A}$.*

$$\Pr[\mathsf{Sim}(\mathsf{H}(\mathsf{Bob}), e, \mathsf{N}) = \mathsf{H}(\mathsf{Bob})^{1/e}] = \frac{\mathsf{a}_{\mathsf{succ}}}{\mathsf{m}} \tag{4.8}$$

Q.e.d.

4.3 SSF with Multiple ID-PKGs

For the multiple ID-PKG case, we follow the proof given by Gennaro et al. who analyzed the SSF protocol [48].

Theorem 4.3.1 (SSF - Multiple ID-PKGs) *Assuming the* RSAA, *the* C-DHA *and the* GAP-DHA *assumptions* SSF *with multiple ID-PKG is a secure, authenticated key agreement protocol in the sense of the CKM, if the hash function* H *is modeled as a programmable, random oracle.*

Proof 4.3.2 ***Case 1: A Matching Test-Session.*** *Consider the problem that the Simulator* Sim *is faced with the problem to solve the* C-DHA *over the composite ring* $\mathbb{Z}_{\mathsf{N}_2}$: $g^{uv} \leftarrow (\mathsf{U} = g^{u}, \mathsf{V} = g^{v}, \mathsf{N}_2)$.

Therefore, he sets the PSP of the first ID-PKG to $\mathsf{D}_1 = \{\mathsf{N}_1, \mathsf{R}_1, \mathsf{G}_1, \mathsf{H}\}$ *with a known factorization of* N_1 *and* R_1, G_1 *and* H *according to the setup algorithm. For the second ID-PKG, he uses the challenge value* N_2 *and a special base:* $\mathsf{D}_2 = \{\mathsf{N}_2, \mathsf{R}_2, \mathsf{G}_2 =$

$\mathsf{g}^{2R_1R_2}, \mathsf{H}\}$. *$R_2$ and H_2 are chosen also according to the setup algorithm. Note, since the factorization of N_2 is unknown, R_1 and R_2 can not be tested to be co-prime to $\varphi(N_2)$. However, a large primes should be fulfill this requirement with overwhelming probability. We denote as $\tau \equiv (2R_1R_2)^{-1} \pmod{\varphi(N_2)}$. As the common basis he uses the CRT to compute $\hat{G}$ from G_1 and G_2. Next,* Sim *computes*

$$\hat{U} = \hat{G}^{\hat{u}} = \begin{cases} G_1^{u'} \pmod{N_1} \\ U \pmod{N_2} = G_2^{\tau u} \pmod{N_2} \end{cases} \tag{4.9}$$

$$\hat{V} = \hat{G}^{\hat{v}} = \begin{cases} G_1^{v'} \pmod{N_1} \\ V \pmod{N_2} = G_2^{\tau v} \pmod{N_2} \end{cases} \tag{4.10}$$

for random values u' and v'. For all parties in the domain of ID-PKG$_1$, Sim *is able to compute the private keys, since he knows the factorization of N_1. For all users in the domain of ID-PKG$_2$ he programs the random oracle to set $H(ID) = r^{R_2} \pmod{N_2}$, thus* Sim *knows also those private keys.*

Sim *guesses that in the i-th protocol execution, $\mathcal{A}$ will eavesdrop the communication and uses the gained information to guess the session key and breaks the* SSF *protocol with success probability a_{succ}.*

We assume that this i-th session takes place between Alice and Bob, with Alice being the initiator. Sim *constructs the Alice's SIK in this round via*

$$\text{SIK}_{Alice} \equiv \hat{U}\hat{d}_{Alice} \pmod{N_1N_2} \tag{4.11}$$

and Bob's SIK is

$$\text{SIK}_{Bob} \equiv \hat{V}\hat{d}_{Bob} \pmod{N_1N_2} \tag{4.12}$$

The session key in this session is $K = H(\hat{G}^{2\hat{u}\hat{v}R_1R_2}) \pmod{N_1N_2}$, which reduces modulo N_2 to

$$K = H(\hat{G}^{2\hat{u}\hat{v}R_1R_2}) \equiv H(G_2^{2\tau\tau uvR_1R_2}) \equiv H(g^{uv}) \pmod{N_2}$$

Among all polynomially bounded oracle queries by the attacker, Sim *finds the query that contains the value g^{uv}.*

Case 2: No Matching Test Session. *In this case, the SIK message can again be tampered by a malicious user. Thus, the two participants will not have a matching session. Like in the single ID-PKG case, we show that even here a protocol breaking* *attacker can be utilized to forge RSA signatures. Suppose $\mathcal{F}$ is faced with the problem to compute a signature (the* e*-th root) out of* $\mathsf{H}(\mathsf{Bob})^{1/\mathsf{e}} \pmod{\mathsf{N}_2}$*. Therefore, he sets the PSP of ID-PKG*$_2$ *to* $\mathsf{D}_2 = \{\mathsf{N}_2, \mathsf{R}_2 = \mathsf{e}, \mathsf{G}_2, \mathsf{H}\}$*. The values for the first ID-PKG he chooses according to the setup algorithm.*

We can assume that $\mathcal{F}$ uses a signing oracle to learn all secret keys of the parties in of ID-PKG$_2$*, except the one of Bob, since $\mathcal{F}$ tries to forge it. Since $\mathcal{F}$ has hence control over all private keys, he can respond to all of the attacker queries unless he is queried about Bob. Next, we show how $\mathcal{F}$ simulates the* i*-th run, that means the session in which $\mathcal{A}$ attacks the* SSF *protocol.*

Simulation of the* i*-th run. *For simplification, we write*

$$\delta_{1,2} \equiv (2\mathsf{R}_1\mathsf{R}_2)^{-1}\mathsf{e}_2 \pmod{\varphi(\mathsf{N}_2)}$$

(which is not known to $\mathcal{F}$ since he does not know the factorization of N_2*). Furthermore, the forger chooses the random integers* r, s *and* f *in* $\mathbb{Z}_{\mathsf{N}_2}$*, where* $\gcd(\mathsf{f}, \mathsf{e}_2) = 1$*. Afterwards, the forger sets*

$$\mathsf{G}_2 \equiv (\mathsf{r}\mathsf{H}(\mathsf{Bob}))^{2\mathsf{R}_1\mathsf{R}_2} \pmod{\mathsf{N}_2} \tag{4.13}$$

and

$$\hat{\alpha} = \begin{cases} \mathsf{G}_2^{\mathsf{x}}\mathsf{H}(\mathsf{Alice}) \pmod{\mathsf{N}_1} \\ (\mathsf{r}\mathsf{H}(\mathsf{Bob}))^{\mathsf{f}} \pmod{\mathsf{N}_2} \end{cases} \tag{4.14}$$

The choice implies that $\hat{\alpha} \equiv \mathsf{G}_2^{\mathsf{x}}\widetilde{\mathsf{d}}_{\mathsf{Alice}} \equiv \mathsf{G}_2^{\delta_{1,2}\mathsf{d}_2\mathsf{f}} \pmod{\mathsf{N}_2}$*, remember that* $\widetilde{\mathsf{d}}_{\mathsf{Alice}} \equiv 1 \pmod{\mathsf{N}_2}$*, thus* $\mathsf{x} \equiv \delta_{1,2}\mathsf{d}_2\mathsf{f} \pmod{\varphi(\mathsf{N}_2)}$*. The attacker now outputs a random session initiation key* β *and its guess for the session key* K*, such that* $\mathsf{K} \equiv \left(\beta^{\mathsf{R}_1\mathsf{R}_2}\mathsf{H}(\mathsf{Bob})^{-1}\right)^{2\mathsf{x}} \pmod{\mathsf{N}_2}$*, which can also be written as*

$$\mathsf{K} \equiv \left(\beta^{\mathsf{e}_2(\mathsf{R}_1\mathsf{R}_2/\mathsf{e}_2)}\mathsf{H}(\mathsf{Bob})^{-\mathsf{R}_1\mathsf{R}_2/\mathsf{e}_2}\right)^{2\delta_{1,2}\mathsf{d}_2\mathsf{f}} \equiv \beta^{\mathsf{e}_2}\mathsf{H}(\mathsf{Bob})^{-\mathsf{d}_2\mathsf{f}} \pmod{\mathsf{N}_2} \tag{4.15}$$

or equivalently

$$\mathsf{K}^{\mathsf{e}_2} \equiv \beta^{\mathsf{e}_2\mathsf{f}}\mathsf{H}(\mathsf{Bob})^{-\mathsf{f}} \pmod{\mathsf{N}_2} \quad \Leftrightarrow \mathsf{H}(\mathsf{Bob})^{\mathsf{f}} \equiv \beta^{\mathsf{e}_2\mathsf{f}}\mathsf{K}^{-\mathsf{e}_2} \pmod{\mathsf{N}_2} \tag{4.16}$$

Now we utilize again that $\gcd(\mathsf{e}_2, \mathsf{f}) = 1$ *and compute* $\mathsf{a}\mathsf{e}_2 + \mathsf{b}\mathsf{f} = 1$*. Afterwards, $\mathcal{F}$*

compute

$$\mathsf{H}(\mathsf{Bob}) = \mathsf{H}(\mathsf{Bob})^{ae_2+bf} = \mathsf{H}(\mathsf{Bob})^{ae_2}\mathsf{H}(\mathsf{Bob})^{bf} \pmod{N_2} \tag{4.17}$$

further

$$\mathsf{H}(\mathsf{Bob})^{ae_2}\mathsf{H}(\mathsf{Bob})^{bf} \pmod{N_2} \equiv (\mathsf{H}(\mathsf{Bob})^{a})^{e_2}\left(\beta^{bf}K^{-b}\right)^{e_2} \pmod{N_2} \tag{4.18}$$

which is equal to $\left(\mathsf{H}(\mathsf{Bob})^{a}\beta^{bf}K^{-b}\right)^{e_2} \pmod{N_2}$*. But this means that*

$$d_{\mathsf{Bob}} \equiv \mathsf{H}(\mathsf{Bob})^{d_2} \equiv \mathsf{H}(\mathsf{Bob})^{a}\beta^{bf}K^{-b} \pmod{N_2} \tag{4.19}$$

Thus, $\mathcal{F}$ *computes the* e_2*-th root of* $\mathsf{H}(\mathsf{Bob})$ *in* $\mathbb{Z}_{N_2}$ *despite he does not know the factorization of* N_2*, which contradicts the RSA assumption.* ***Q.e.d***

4.4 SSF Signatures

In this section, the proofs regarding the signature schemes are presented. The proofs use the *random oracle model* and show a reduction to the RSA assumption. The proofs build upon the approach of the signature proof presented by Gennaro et al. [47]. The proofs cover the strong case, meaning that the scheme is secure against existential forgery on adaptively chosen message and ID attacks.

4.4.1 Single Signature

Theorem 4.4.1 (Basic Version) *Let the* $\mathsf{PSP} = (N, G, R, H)$ *be the public shared parameters and let the output of* H *be a* w*-bit integer with* $2^w < R = \nu \cdot \hat{R}$*. If there is a forger algorithm* $\mathcal{F}$ *that wins the* $\mathsf{ID-CM}$ *game with non-negligible probability* a_0*, then there also exists an adversary* $\mathcal{A}$ *that breaks the RSA assumption with non-negligible probability of*

$$\Pr\left[\begin{array}{c} r^e \equiv t \pmod{N}; (N,e,d) \leftarrow \mathsf{RSAgen}; \\ t \overset{rand}{\leftarrow} \mathbb{Z}_N^*; r \leftarrow \mathcal{A}(N,t) \end{array}\right] > \frac{a_0}{4q_{H_2}} \tag{4.20}$$

Proof 4.4.2 *In the* $\mathsf{ID-CM}$ *game above, the adversary does not possess the private keys since its goal is to solve the RSAA. Furthermore, the adversary is not allowed to* *ask adaptive queries to the extraction oracle, which would give the adversary access to a solution to the RSAA, since RSA is not secure against adaptive adversaries. The adversary only learns the private keys for a random integer, which are independent of the RSAA instance the adversary has to solve.*

The adversary assumes that the forger $\mathcal{F}$ will output its forgery using the j-th identity, which was used as the input in the j-th hash query to H_2. We also define that the adversary makes two times more H_2 queries than the maximum of the extract or sign queries. The adversary makes the following preparation steps.

Phase 1: Preparation. *$\mathcal{A}$ prepares for potential hash queries. Therefore, $\mathcal{A}$ chooses the set of random integers $(e_1, ..., e_{q_{H_1}})$ that are used for the answers to message hash queries made to H_1. As a second set, $\mathcal{A}$ chooses the random integers $(f_1, ..., f_j = t, ..., f_{q_{H_2}})$ that it uses as answers for identity hash queries made to H_2. At position j, it contains the random number t that is part of the RSAA instance $\mathcal{A}$ has to solve. Since all integer are random, they are independent of each other.*

Whenever H_1 or H_2 receives a query, they maintain lists L_1 and L_2 that store the tuples (m_i, e_i) and (ID_i, f_i) respectively.

Phase 2: Query. *In the i-th message hash query, H_1 answers with e_i and in the i-th identity hash query H_2 answers with f_i. If H_2 receives an extract query for an identity ID, H_2 checks if ID maps to one tuple in L_2. If not, H_2 ignores the query. If H_2 finds a matching entry, say (ID_l, f_l) with $ID = ID_l$, H_2 checks if $l = j$. In this case, H_2 **aborts**, since it would have to ask the extraction oracle for the R-th root of $f_j = t$. Otherwise, $\mathcal{A}$ relays the answer from the extraction oracle to $\mathcal{F}$. When receiving a signature query on a message m and for an identity ID, H_2 checks if both elements are part of the lists L_1 and L_2. If not, $\mathcal{A}$ ignores the query. Again, if $ID = ID_j$, H_2 **aborts**. Otherwise, H_2 answers honestly with a signature according to Algorithm 15.*

Phase 3: Guess and Verification. *Suppose in the j-th round, $\mathcal{F}$ tries the forgery and gives the solution (s_1, s_2, m_j, ID_j) using the message hash value $e_j = H(m_j)$ and the identity hash value $t = H(ID_j)$. If the signature is incorrect, $\mathcal{A}$ **aborts**. Otherwise, the signature (s_1, s_2, m_j, ID_j) of the forger $\mathcal{F}$ can be used to solve the RSAA, since $s_1^R s_2^{-\nu e_j} \equiv t \equiv H(ID_j) \pmod{N}$. Since $\nu | R$, the adversary computes*

$$\left(s_1^{\hat{R}} s_2^{-e_j}\right)^{\nu} \equiv H(ID_j) \equiv t \pmod{N} \tag{4.21}$$

and thus recovers a solution $(s_1^{\hat{R}} s_2^{-e_j} = r, \nu = e)$ for the given RSAA instance (N, t).

The probability that $\mathcal{A}$ does not abort, but solves the RSAA successfully needs to be calculated. Since j is chosen randomly from the q_{H_2} integers that are used as the response for an identity hash query, the chance that $\mathcal{F}$ chooses the j-th identity for its forgery is $1/q_{H_2}$. Furthermore, $\mathcal{A}$ aborts either if $\mathcal{F}$ asks for an extract query on $f_j = H(ID_j)$ or $\mathcal{F}$ asks for a signature query regarding the identity ID_j. We assumed that $q_{H_2} > \max(q_E, q_S)/2$. Thus, the probability that $\mathcal{F}$ does not pick f_j during its q_E extract and q_S sign queries is $\geqslant 1/4$. And since the success rate of a valid forgery is a_0, the total probability to break the RSAA is $\geqslant a_0/(4q_{H_2})$, as demanded.

At this point, we have shown that the basic version can be reduced to the RSA assumption, by simply setting $H(ID_j) = t$. In the multi-signer case, the adversary has to choose the responses to the H_2 queries more carefully, as described below.

4.4.2 Multi-Signatures

Theorem 4.4.3 (Multi-Signatures) *Let the $PSP = (N, G, R, H)$ be the public shared parameters and let the output of H be a w-bit integer with $2^w < R = v \cdot \hat{R}$. If there is a forger algorithm $\mathcal{F}$ that wins the $ID - CM$ game by forging a n-multi-signature with non-negligible probability a_0, then there also exists an adversary $\mathcal{A}$ that breaks the RSA assumption with non-negligible probability of*

$$\Pr\left[\begin{array}{r} r^e \equiv t \pmod{N}; (N, e, d) \leftarrow \mathsf{RSAgen}; \\ t \overset{rand}{\leftarrow} \mathbb{Z}_N^*; r \leftarrow \mathcal{A}(N, t) \end{array}\right] > \frac{a_0}{4} - \frac{a_0(q_{H_2} - n)}{4q_{H_2}} \tag{4.22}$$

Proof 4.4.4 *The forger outputs its forgery using n identities. The adversary assumes that the j-th identity that was used as the input in the j-th hash query to H_2, will be among these n identities. Also, we define that the adversary makes two times more H_2 query than the maximum of the extract or sign queries. The adversary makes the following preparation steps.*

Phase 1: Preparation. *$\mathcal{A}$ prepares for potential hash queries. Therefore, $\mathcal{A}$ chooses the set of random integers $(e_1, ..., e_{q_{H_1}})$ that are used for the answers to message hash queries made to H_1. As a second set, $\mathcal{A}$ chooses the random integers $(f_1^v, ..., f_j = t \cdot 2^v, ..., f_{q_{H_2}}^v)$, with f_i random, which $\mathcal{A}$ uses as answers for identity hash queries made to H_2. Note that since the f_i are independent and v is co-prime to $\varphi(N)$, the f_i^v keep their independence and random character.*

Whenever H_1 *or* H_2 *receive a query, they maintain lists* L_1 *and* L_2 *that store the tuples* (m_i, e_i) *and* (ID_i, f_i) *respectively.*

Phase 2: Query. *Equal to Phase 2 in the basic version.*

Phase 3: Guess and Verification. *Suppose in the* j*-th round,* $\mathcal{F}$ *tries the forgery and gives the solution* $(s_1, s_2, m_j, \{\mathsf{ID}_1, ...\mathsf{ID}_n\})$ *using the message hash value* $e_j = \mathsf{H}(m_j)$*. If the signature is incorrect,* $\mathcal{A}$ ***aborts****. Otherwise, the signature of the forger* $\mathcal{F}$ *can be used to solve the RSAA since it holds* $s_1^{\mathsf{R}} s_2^{-\nu e_j} \equiv 2^{\nu} \cdot t \cdot \prod f_k^{\nu} \pmod{\mathsf{N}}$*. Since* $\nu | \mathsf{R}$*, the adversary computes*

$$\left(s_1^{\hat{\mathsf{R}}} s_2^{-e_j} (2 \prod^{n} f_k)^{-1} \right)^{\nu} \equiv t \pmod{\mathsf{N}} \tag{4.23}$$

and thus recovers a solution $(s_1^{\hat{\mathsf{R}}} s_2^{-e_j} (2 \prod^n f_k)^{-1} = r, \nu = e)$ *for the given RSAA instance* (N, t)*.*

The probability that $\mathcal{A}$ *does not abort, but solves the RSAA successfully needs to be calculated. Since* j *is chosen randomly from the* q_{H_2} *integers that are used as the response for an identity hash query, the chance that* $\mathcal{F}$ *chooses the* j*-th identity for its forgery made of* n *identities is* $1 - \prod_{j=0}^{n-1} \frac{q_{\mathsf{H}_2} - j - 1}{q_{\mathsf{H}_2} - j} = 1 - \frac{q_{\mathsf{H}_2} - n}{q_{\mathsf{H}_2}}$*. Furthermore,* $\mathcal{A}$ *aborts either if* $\mathcal{F}$ *asks for an extract query on* $f_j = \mathsf{H}(\mathsf{ID}_j)$ *and or* $\mathcal{F}$ *asks for a signature query regarding the identity* ID_j*. We assumed that* $q_{\mathsf{H}_2} > \max(q_{\mathsf{E}}, q_{\mathsf{S}})/2$*. Thus, the probability that* $\mathcal{F}$ *does not pick* f_j *during its* q_{E} *extract and* q_{S} *sign queries is* $\geqslant 1/4$*. And since the success rate of a valid forgery is* a*, the total probability to break the RSAA is* $\geqslant \mathsf{a}_0/4 - \mathsf{a}_0(q_{\mathsf{H}_2} - n)/(4 q_{\mathsf{H}_2})$ *as demanded.*

4.4.3 Multi-Signatures with Multiple ID-PKGs

We now include multiple ID-PKGs. In this scenario, we show that a forged signature would lead to a forged signature in the basic case and thus breaks the RSA assumption.

Theorem 4.4.5 (Multi-Signatures with multiple ID-PKG) *Let the* $\mathsf{PSP} = (\mathsf{N}, \mathsf{G}, \mathsf{R}, \mathsf{H})$ *be the public shared parameters and let the output of* H *be a* w*-bit integer with* $2^w < \mathsf{R} = \nu \cdot \hat{\mathsf{R}}$*. If there is a forger algorithm* $\mathcal{F}$ *that wins the* $\mathsf{ID} - \mathsf{CM}$ *game by forging a* n*-multi-signature with* w *independent ID-PKGs with non-negligible probability* a_0*, then there also exists an adversary* $\mathcal{A}$ *that breaks the RSA assumption with non-negligible* *probability of*

$$\Pr\left[\begin{matrix} r^e \equiv t \pmod{N_k}; (N_k, e, d) \leftarrow \mathsf{RSAgen}; \\ t \overset{rand}{\leftarrow} \mathbb{Z}_N^*; r \leftarrow \mathcal{A}(N,t) \end{matrix}\right] \geqslant \frac{a_0}{w4} - \frac{a_0(q_{H_2} - n)}{w4q_{H_2}} \tag{4.24}$$

Proof 4.4.6 *In this case, the game between the adversary and the forger is similar to the previous cases. The difference is the existence of several PSPs, which leads to a change of the extract queries. The forger now has to additionally specify the modulus the identity key is valid for. Thus, the* L_2 *list keeps entries of the form* $(\mathsf{ID}_i, f_i, N_i)$*. The forger is equipped with all involved* w *PSPs, where PSP*$_k$ *contains the integer* N_k *that is the target modulus for the adversary regarding the RSAA. The adversary assumes that the forger outputs its forgery using* n *different identities. It always holds that* $w \leqslant n$*, and the equality occurs whenever each signer comes from a unique ID-PKG. The adversary further assumes that the* j*-th identity that was used as the input in the* j*-th hash query to* H_2 *will be among these identities. Also, we define that the adversary makes two times more* H_2 *queries than the maximum of the extract or sign queries. The adversary makes the following preparation steps.*

Phase 1: Preparation. $\mathcal{A}$ *prepares for potential hash queries. Therefore,* $\mathcal{A}$ *chooses the set of random integers* $(e_1, ..., e_{q_{H_1}})$ *that are used for the answers to message hash queries made to* H_1 *and* $(f_1^\nu, ..., f_j = t \cdot 2^\nu, ..., f_{q_{H_2}}^\nu)$ *that are used as answers for identity hash queries made to* H_2*. Whenever* H_1 *or* H_2 *receive a query, they maintain lists* L_1 *and* L_2 *that store the tuples* (m_i, e_i) *and* (ID_i, f_i)*, respectively.*

Phase 2: Query. *Equal to Phase 2 in the basic version.*

Phase 3: Guess and Verification. *Suppose in the* j*-th round,* $\mathcal{F}$ *tries the forgery and gives the solution* $(s_1, s_2, m_j, \{\mathsf{ID}_1, ...\mathsf{ID}_n\})$ *using the message hash value* $e_j = H(m_j)$*. If the signature is incorrect,* $\mathcal{A}$ ***aborts****. At this point, the adversary only cares about the integer* N_k*. The adversary tests if* N_k *is part of the multi-modulus by a simple GCD computation. If it is not part of the product,* $\mathcal{A}$ ***aborts****. If* $\mathcal{A}$ *finds* N_k *as a factor of the modulus, it obtains the congruence*

$$\left(s_1^{\hat{R}} s_2^{-e_j}\right)^\nu \equiv X \pmod{\prod_{i=1}^{w} N_i}, \tag{4.25}$$

where X *is the product of the extended hash values received from the* $\mathsf{HashValExt}$ *algorithm. If* u *is the number of signers that are associated with* N_k*, then* $X \equiv 2^\nu t \prod_{l \neq j}^{u} f_l^\nu \pmod{N_k}$ *whenever* $\mathcal{F}$ *associated the* j*-th identity with the* k*-th moduli. Thus,* $\mathcal{A}$ *finds* *a solution to the RSAA that is* $\left(s_1^{\hat{R}} s_2^{-e_j} (2 \prod_{l \neq j}^{u} f_l)^{-1} = r, e = \nu\right)$.

The probability that $\mathcal{A}$ *does not abort, but solves the RSAA successfully needs to be calculated. Since* j *is chosen randomly from the* q_{H_2} *integers that are used as the response for an identity hash query, the chance that* $\mathcal{F}$ *chooses the* j*-th identity for its forgery made of* n *identities is* $1 - \prod_{j=0}^{n-1} \frac{q_{H_2}-j-1}{q_{H_2}-j} = 1 - \frac{q_{H_2}-n}{q_{H_2}}$. *Assigning* ID_j *to the integer* N_k *out of* w *possible is about* $\frac{1}{w} - \frac{q_{H_2}-n}{wq_{H_2}}$, *which yields a total probability that* $\mathcal{A}$ *does not abort of* $\geqslant a_0/(w4) - a_0(q_{H_2} - n)/(w4q_{H_2})$ *as demanded.*

4.5 Summary

Chapter 4 presented the security proofs of the proposed scheme. The security of the key agreement protocol was proven in both scenarios; in the single ID-PKG case as well as in the multiple ID-PKG case. For a key agreement scheme it is sufficient to prove the case of two involved ID-PKGs, since a key agreement always only takes place between two participants. For the proof, the Canetti-Krawczyk Model was used; it is one of the standard models to define the requirements for a secure and authenticated key agreement protocol. If there is an attacker that successfully breaks the SSF protocol with non-negligible probability, it was shown that this is sufficient to construct a scenario where this attacker can be used as a subroutine to break the computational Diffie-Hellman assumption or the RSA assumption. This is contradictory to the common believe that these two problems can not be solved in non-negligible time.

For the proof of the proposed signature scheme, an adaptive adversary was taken, which tries to forge a signature for an arbitrary identity. Three cases were distinguished: A single signature with one ID-PKG, a multi-signature with one ID-PKG and a multi-signature with multiple ID-PKGs. In all three cases, it was shown via reduction that an adversary that can forge a signature with non-negligible probability is also able to break the RSA assumption with non-negligible probability.

5 Related Attacks

"Human ingenuity cannot concoct a cipher which human ingenuity cannot resolve."
Edgar Allan Poe

5.1 Introduction

In this chapter, attacks that are related to the presented scheme are discussed. These attacks do not apply to the presented scheme directly (note that its security was already proven in the CK-Model), but to common extensions that could be applied to the presented scheme. The first attack applies when the modulus is changed from $\mathsf{N} = \mathsf{PQ}$ to $\mathsf{N} = \mathsf{PQ}^{2e}$. This attack is connected with the Φ-Hiding assumption. The second attack is about *Secret Sharing Schemes*. Such schemes are used to distribute a secret among a set of users who are not allowed to get the knowledge about the entire secret, but only parts of it. Boneh and Franklin [22] proposed an approach to generate the identity keys of each user to reduce the *key escrow* problem. In this chapter, it will be shown that some of these secret sharing schemes are insecure when using them to share the integer $\varphi(\mathsf{N})$, which would be exactly the case when applying this approach to the presented scheme.

The results of the attacks are described in [104] and [105].

5.2 The Φ-Hiding Assumption

In Chapter 4, the security of the proposed scheme regarding existing models from the literature was proven. It was shown that SSF is a secure and authenticated key agreement protocol and is secure against existential forgery on adaptively chosen message and ID attacks.

However, there are also other ways to weaken the scheme, e.g., by getting information about the identity keys, that is by learning something about the integer $\varphi(N)$. A related assumption is the Φ-Hiding assumption, as defined by Cachin, Micali and Stadler[27]. It is about the difficulty to decide if a given integer is a divisor of $\varphi(N)$ or not, where N is a number whose factorization is unknown (and cannot be computed). The security of several cryptosystems is based on the presumed difficulty of solving this problem [26, 49, 50, 61].

The Φ-Hiding assumption is a stronger assumption than the integer factorization problem and it looks on the first sight pretty clear to be secure as long as the IFP resists cryptanalysis. The IFP states that $\varphi(N)$ is hard to compute if N is a larger integer; the PHA states that $\varphi(N)$ is not only hard to compute, but it is already infeasible to decide if a given integer is a factor of $\varphi(N)$ or not. Obviously, their exists a trivial case, namely the integers 1 and 2 will always divide $\varphi(N)$ and are thus excluded from this assumption.

In the sequel it is shown that this glance of the PHA is wrong. It will be shown that, despite the factorization of N is unknown, there can be gained information about $\varphi(N)$ if N is of the form $N = PQ^{2e}$, where $P, Q > 2$ are primes, $e > 0$ is an integer and P hides the prime in question. This information can lastly help to break the PHA under the named circumstances. Moduli of the form $N = PQ^{2e}$ are not exceptional or abnormal. These moduli are called *Multi-Power RSA* moduli and are used to speed up cryptographic operations. Boneh [22] illustrates in a short survey the speedup when using this kind of integers. In addition, it will be shown that if the PHA is instantiated that a random composite integer is hidden instead of a prime, the probability of choosing the integer that divides $\varphi(N)$ reaches 99% if the integer has at least 7 prime factors.

In Chapter 2, the PHA was defined in its special form. Next, the PHA is redefined, once in its plain version and once in the same way as in Section 2 for a better comparability. The first definition illustrates the computational problem the assumption is based on.

Definition 5.2.1 (Φ-Hiding assumption (1)) *Given an integer N with unknown factorization, it is computationally hard to decide whether a prime p_i with $2 < p_i << N^{1/4}$ divides $\varphi(N)$ or not.*[1]

[1]Following the remarks of the original paper of Cachin, Micali and Stadler [27], N can be efficiently factored when a prime $> N^{1/4}$ of $\varphi(N)$ is known, thus the Φ-Hiding assumption asks for very small primes. Even if it is known which small primes p_i divide $\varphi(N)$, if $\log\ p_i$ is significantly smaller

The second definition represents a special case of the assumption, since it is assumed that exactly one of two given integers divides $\varphi(N)$.

Definition 5.2.2 (Φ-Hiding assumption (2)) *Let* $p_1 > 2$ *and* $p_2 > 2$ *be two random, small primes and* N *be an integer that is constructed such that exactly one of these two primes divides* $\varphi(N)$*. Then for any probabilistic polynomial time adversary* $\mathcal{A}$ *, the advantage function, if* p_b *divides* $\varphi(N)$*,* $b \in \{1, 2\}$

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{PHA}} = \left| \Pr[\mathcal{A}(N, p_1, p_2) = b] - \frac{1}{2} \right|,$$

is negligible.

In cryptographic protocols, Definition 5.2.2 of the Φ-Hiding assumption is used, since in this case some previous knowledge is involved (i.e. which of the two primes divides $\varphi(N)$), that can be used to create a necessary backdoor for asymmetric cryptography. To the best of our knowledge, this is the first attack on the Φ-Hiding assumption until now.

5.3 The Φ-Hiding Assumption Revisited

The Φ-Hiding assumption is only valid when it is applied to a composite number that cannot be completely factored in feasible time, since otherwise it would be trivial to decide whether a prime divides $\varphi(N)$ or not. The proposed approach to decide whether a prime divides $\varphi(N)$ for a composite number N uses the Jacobi symbol. Furthermore, a particular $2k$-th root of unity is used to show that the values of the Jacobi symbol are related to factors of $\varphi(N)$, and that the Jacobi symbol adopts *non-random* values when the evaluated integer r is a divisor of $\varphi(N)$. Thus, the novel idea to use the existence and the non-existence of $2k$-th roots of unity in finite fields/rings allows to gain knowledge about the divisors of $\varphi(N)$, which in some cases can be used to make the decision whether a given integer divides $\varphi(N)$ or not. These results will be used to show that the Φ-Hiding assumption as defined by Cachin, Micali and Stadler [27] is not valid when applied to a modulus $N = PQ^{2e}$, where $P, Q > 2$ are primes, $e > 0$ is an integer and P hides the prime in question.

Next, the first Lemma 5.3.1 is defined, which is central for the approach:

than $(\log\ N)^c$, for a constant c between 0 and 1, N cannot be factored significantly faster.

Lemma 5.3.1 *Let ξ_{2k} be any fixed primitive $2k$-th root of unity and $k \in \mathbb{N}^+$, then:*

$$i^{1-k} \prod_{j=1}^{k-1} \left(\xi_{2k}^{j} - \xi_{2k}^{-j} \right) = k \tag{5.1}$$

Proof 5.3.2 (of Lemma 5.3.1) *The polynomial $f(X) = (X^k - 1)/(X - 1) = X^{k-1} + X^{k-2} + ... + 1$ has ξ_k^j for $j = 1, ..., k-1$ as its roots, where ξ_k is any fixed primitive kth root of unity. Writing $f(X)$ in factored form $f(X) = \prod_{j=1}^{k-1}(X - \xi_k^j)$, we obtain $f(1) = \prod_{j=1}^{k-1}(1 - \xi_k^j) = k$. Since*

$$i^{1-k} \prod_{j=1}^{k-1} (\xi_{2k}^{j} - \xi_{2k}^{-j}) = i^{1-k} \prod_{j=1}^{k-1} \xi_{2k}^{j} \prod_{j=1}^{k-1} (1 - \xi_{k}^{-j}) = i^{1-k} k \prod_{j=1}^{k-1} \xi_{2k}^{j} \tag{5.2}$$

and since $\prod_{j=1}^{k-1} \xi_{2k}^{j} = \xi_{2k}^{(k-1)k/2} = \xi_4^{k-1} = i^{k-1}$, the product $i^{1-k} \prod_{j=1}^{k-1} \xi_{2k}^{j}$ vanishes and we get

$$i^{1-k} \prod_{j=1}^{k-1} (\xi_{2k}^{j} - \xi_{2k}^{-j}) = k \tag{5.3}$$

which proves the lemma. □

The $(k - 1)$ terms, covered by the product symbol in equation (5.1), can be rewritten such that it contains a large square:

Lemma 5.3.3 (Square Lemma) *Let $k \in \mathbb{Z}^+$ and $k > 2$. Then:*
1. If k is odd:

$$\prod_{j=1}^{k-1} \left(\xi_{2k}^{j} - \xi_{2k}^{-j} \right) = \prod_{j=1}^{(k-1)/2} \left(\xi_{2k}^{j} + \xi_{2k}^{k-j} \right)^2 \tag{5.4}$$

2. If k is even:

$$\prod_{j=1}^{k-1} \left(\xi_{2k}^{j} - \xi_{2k}^{-j} \right) = 2i \prod_{j=1}^{(k-2)/2} \left(\xi_{2k}^{j} + \xi_{2k}^{k-j} \right)^2 \tag{5.5}$$

Proof 5.3.4 (of Lemma 5.3.3)
1. k is odd: *Since k is odd, the jth and the $(k - j)$th factor for $1 \leqslant j \leqslant k - 1$ can be paired. The result is:*

$$(\xi_{2k}^{j} - \xi_{2k}^{-j}) \cdot (\xi_{2k}^{k-j} - \xi_{2k}^{-(k-j)}) = (\xi_{2k}^{j} - \xi_{2k}^{-j}) \cdot (\xi_{2k}^{k-j} + \xi_{2k}^{j})$$

$$\begin{aligned}
&= \xi_{2k}^{j}\xi_{2k}^{k-j} + \xi_{2k}^{j}\xi_{2k}^{j} - \xi_{2k}^{-j}\xi_{2k}^{k-j} - \xi_{2k}^{-j}\xi_{2k}^{j} = -1 + \xi_{2k}^{2j} - \xi_{2k}^{k-2j} - 1\\
&= \xi_{2k}^{2j} - 2 - \xi_{2k}^{k-2j} = \xi_{2k}^{2j} - 2 + \xi_{2k}^{k}\xi_{2k}^{k-2j}\\
&= \xi_{2k}^{2j} - 2 + \xi_{2k}^{2(k-j)} = (\xi_{2k}^{j} + \xi_{2k}^{k-j})^2
\end{aligned}$$

The pairing contains a square. Since $k-1$ *is even, no term is left and a product of* $(k-1)/2$ *squares is generated, which proves the case for odd values of* k.
2. <u>k is even:</u> Since k *is even, the* j*th and the* $(k-j)$*th factor for* $1 \leqslant j < k/2$ *and* $k/2 < j \leqslant k-1$ *can be paired, which leads to the same terms as in case 1. The difference is that the factor* $\left(\xi_{2k}^{j} - \xi_{2k}^{-j}\right)$ *with* $j = k/2$ *remains. For this factor,* $\xi_{2k}^{k/2} - \xi_{2k}^{-k/2} = (-1)^{1/2} - (-1)^{-1/2} = i - i^{-1} = i(1 - 1/i^2) = 2i$, *which proves the case for even values of* k. □

By Lemma 5.3.3, the product in equation (5.1) is transformed to a product with a perfect square and the factor i^{1-k} (k odd) and $2i^{2-k}$ (k even), respectively. Square numbers play an important role in cryptography, just when operating in a ring $\mathbb{Z}_N$, with N of unknown factorization. Computing square roots is a one-way function in such rings, even more, to decide if an integer actually has a square root is already infeasible. However, cryptologists have access to the Jacobi symbol that decides for some integers correctly if they have a square root in the ring or not, even if the factorization is unknown. Integers that are already a square number, like the developed term in the lemma above, are thus ignored by the Jacobi-symbol since they have already an integer square root in $\mathbb{Z}$, which make the Jacobi symbol always equal to one.

5.3.1 Application to Finite Fields and Rings

In this section, the results are applied to finite fields $\mathbb{F}_P$ with P being a prime number. It is distinguished between two cases. In the first case, it is assumed that a $\xi_{2k} \in \mathbb{F}_P$ does not exist, and in the second case, it is assumed that a $\xi_{2k} \in \mathbb{F}_P$ exists.

5.3.1.1 Case 1: A $\xi_{2k} \in \mathbb{F}_P$ does not exist.

In this case, it is assumed that $\mathbb{F}_P$ does not contain a $2k$-th root of unity. As a consequence, there is no integer of order $2k$ and thus the factors $\left(\xi_{2k}^{j} + \xi_{2k}^{k-j}\right)$ are not defined properly in $\mathbb{F}_P$. Thus, it cannot be assumed that the product $\prod_{j=1}^{(k-1)/2}\left(\xi_{2k}^{j} + \xi_{2k}^{k-j}\right)^2$

forms a valid square in $\mathbb{F}_P$ and vanishes from the Jacobi symbol. The integer k, which nevertheless exists, has no defined counterpart on the left side of equation 5.1. In this case, $J_P(k)$ cannot be distinguished from a random coin flip between 1 and -1.

5.3.1.2 Case 2: A $\xi_{2k} \in \mathbb{F}_P$ exists.

This leads to the fact that the square $\prod_{j=1}^{(k-1)/2} \left(\xi_{2k}^{j} + \xi_{2k}^{k-j}\right)^2$ obtained from Lemma 5.3.3 is valid in $\mathbb{F}_P$, since each ξ_{2k} is defined properly. Therefore, equation (5.1) can be written as a well defined congruence in $\mathbb{F}_P$. Corollary 5.3.5 shows the outcome when the Jacobi symbol is applied to this congruence and the square obtained from Lemma 5.3.3 is inserted.

Corollary 5.3.5 *Let P be an odd prime number, $k \in \mathbb{F}_P$. Assume that a $\xi_{2k} \in \mathbb{F}_P$ exists, then:*
1. If k is odd:

$$J_P\left((-1)^{(1-k)/2} \prod_{j=1}^{(k-1)/2} \left(\xi_{2k}^{j} + \xi_{2k}^{k-j}\right)^2\right) = J_P((-1)^{(1-k)/2}) = J_P(k) \tag{5.6}$$

2. If k is even:

$$J_P\left(2(-1)^{1-k/2} \prod_{j=1}^{(k-2)/2} \left(\xi_{2k}^{j} + \xi_{2k}^{k-j}\right)^2\right) = J_P(2(-1)^{1-k/2}) = J_P(k) \tag{5.7}$$

After the square has vanished from the Jacobi symbol, a simple congruence is left. This congruence indicates a relationship between the value of the Jacobi symbol and the divisors of $\varphi(P)$, because Corollary 5.3.5 is only valid if $2k$ divides $\varphi(P)$. Again, this implicitly shows that it is important to distinguish between the two cases of divisibility introduced above, since the square vanishes only if it is defined properly. Otherwise, the Jacobi symbol of an arbitrary integer k would always be equal to $J_P((-1)^{(1-k)/2})$ or $J_P(2(-1)^{1-k/2})$, respectively, which obviously is wrong.

EXAMPLE: Let $P = 31$ with $\varphi(31) = 30$. By setting $k = 5$ due to $(2 \cdot 5)|30$, there must be an integer of order 10, e.g. 23 or 15. It does not matter which of them is chosen here, since it disappears after applying the Jacobi symbol. Now, calculate $(-1)^{(1-5)/2} = (-1)^{-2} = 1$. Since k is odd, $J_{31}((-1)^{(1-5)/2}) = J_{31}(1) = J_{31}(5)$ must hold, which is true since both sides are equal to 1.

Next, a theorem is stated that describes the relationship between $J_P(k)$ and ξ_{2k}.

Theorem 5.3.6 *Let P be an odd prime number, $k \in \mathbb{F}_P$. $J_P(k)$ and the divisors of $\varphi(P)$ are connected via following implications:*
1. If k is odd, then:

$$\begin{array}{rcl} \textit{If } \xi_{2k} \in \mathbb{F}_P \textit{ exists} & \Rightarrow & J_P((-1)^{(1-k)/2}) = J_P(k). \\ \textit{If } J_P((-1)^{(1-k)/2}) \neq J_P(k) & \Rightarrow & \xi_{2k} \in \mathbb{F}_P \textit{ does not exist.} \end{array}$$

2. If k is even, then:

$$\begin{array}{rcl} \textit{If } \xi_{2k} \in \mathbb{F}_P \textit{ exists} & \Rightarrow & J\left(2(-1)^{1-k/2}\right) = J_P(k). \\ \textit{If } J\left(2(-1)^{1-k/2}\right) \neq J_P(k) & \Rightarrow & \xi_{2k} \in \mathbb{F}_P \textit{ does not exist.} \end{array}$$

Proof 5.3.7 (of Theorem 5.3.6)
The proof of the theorem follows directly from Corollary 5.3.5. □

Theorem 5.3.6 indicates that either a divisor k of $\varphi(P)$ must be known to conclude that the corresponding Jacobi symbols $J_P(k)$ and $J_P((-1)^{(1-k)/2})$ (or $J\left(2(-1)^{1-k/2}\right)$) are equal, or it must be tested whether the two Jacobi symbols $J_P(k)$ and $J_P((-1)^{(1-k)/2})$ (or $J\left(2(-1)^{1-k/2}\right)$) are different in order to get the information that k cannot be a divisor of $\varphi(P)$. In the two other cases, no information can be obtained. The reason is that either the kth root of -1 is not defined, or from the equality of the Jacobi symbols it cannot be concluded that k divides $\varphi(P)$.

To summarize, if $2k$ divides $\varphi(P)$, the Jacobi symbol of k adopts non-random values. Furthermore, Corollary 5.3.5 shows that the resulting congruences $J_P((-1)^{(1-k)/2}) \equiv J_P(k)$ and $J_P(2(-1)^{1-k/2}) \equiv J_P(k)$ for odd and even values of k are *independent* of the chosen ξ_{2k}. Thus, it is only essential that a ξ_{2k} exists in $\mathbb{F}_P$, but it is not necessary to know them.

5.3.2 Leakage Corollaries

In this section, tables for special composite integers N are presented that contain the values the Jacobi symbol must adopt to leak information about the divisors of $\varphi(N)$. For composite integers N with unknown factorization, the order of an arbitrary integer a is not known, but one can compute the Jacobi symbol $J_N(a)$. Thus, only the first implication of item 1 and and the second implication of item 2 of Theorem 5.3.6 can be used. For clarity, the following Corollary divides these items further with respect to different residue classes of a prime P and an integer k.

Corollary 5.3.8 (Leakage Corollary for prime numbers) *Let P be an odd prime number, $\mathsf{k} \in \mathbb{F}_{\mathsf{P}}$. In any of the following six cases, there does not exist a $\xi_{2\mathsf{k}} \in \mathbb{F}_{\mathsf{P}}$.*
If $\mathsf{P} \equiv 1 \pmod 4$:

> *If k is odd: If $J_{\mathsf{P}}\left(\mathfrak{i}^{1-\mathsf{k}}\right) = 1 \neq -1 = J_{\mathsf{P}}(\mathsf{k})$.*
> *If k is even: If $J_{\mathsf{P}}\left(2\mathfrak{i}^{2-\mathsf{k}}\right) = (-1)^{(p^2-1)/8} \neq J_{\mathsf{P}}(\mathsf{k})$.*

If $\mathsf{P} \equiv 3 \pmod 4$:

> *If $\mathsf{k} \equiv 0 \pmod 4$: If $J_{\mathsf{P}}\left(2(-1)^{1-\mathsf{k}/2}\right) = (-1)^{(\mathsf{P}^2+7)/8} \neq J_{\mathsf{P}}(\mathsf{k})$.*
> *If $\mathsf{k} \equiv 1 \pmod 4$: If $J_{\mathsf{P}}\left((-1)^{(1-\mathsf{k})/2}\right) = 1 \neq J_{\mathsf{P}}(\mathsf{k})$.*
> *If $\mathsf{k} \equiv 2 \pmod 4$: If $J_{\mathsf{P}}\left(2(-1)^{1-\mathsf{k}/2}\right) = (-1)^{(\mathsf{P}^2-1)/8} \neq J_{\mathsf{P}}(\mathsf{k})$.*
> *If $\mathsf{k} \equiv 3 \pmod 4$: If $J_{\mathsf{P}}\left((-1)^{(1-\mathsf{k})/2}\right) = -1 \neq J_{\mathsf{P}}(\mathsf{k})$.*

The Corollary states which two Jacobi symbols must differ to be sure that the integer k is not a divisor of $\varphi(\mathsf{P})$. Thus, in some cases, the access to the Jacobi symbol is sufficient to decide whether a prime divides $\mathsf{P} - 1$ or not. Next, the Corollary is extended to composite integers N being the product of two distinct prime numbers P and Q. This leads to the tables shown in Figure 5.1.

The tables must be read in the following way: The four tables handle the four different residues of k modulo 4. Furthermore, the first two tables (horizontal direction) show the 64 combinations of the 8 different residues of P and Q modulo 16 ($\mathsf{P}, \mathsf{Q} > 2$) for even residues of k. The third tables was reduced to one a single row since it contains 64 values of -1. The fourth table shows the 64 combinations of the 8 different residues of P and Q modulo 16 ($\mathsf{P}, \mathsf{Q} > 2$) for $\mathsf{k} \equiv 3 \pmod 4$. The entries for each combination of P and Q illustrate which value of the Jacobi symbol $J_{\mathsf{N}}(\mathsf{k})$ reveals that there is no integer of order $2\mathsf{k}$ for at least one of the primes P and Q. For example, the first entry of -1 in the upper left table represents the case $\mathsf{k} \equiv 0 \pmod 4$ and $\mathsf{P} \equiv \mathsf{Q} \equiv 1 \pmod{16}$. Applying Corollary 5.3.8 to this combination yields $J_{\mathsf{P}}\left(2\mathfrak{i}^{2-\mathsf{k}}\right) = J_{\mathsf{Q}}\left(2\mathfrak{i}^{2-\mathsf{k}}\right) = 1$. The corresponding table entry of -1 shows that $J_{\mathsf{N}}(\mathsf{k})$ must be -1, therefore at least for one of the primes P or Q, there is no integer of order $2\mathsf{k}$.

The conclusion is too weak to obtain knowledge regarding the Φ-Hiding assumption, since $\phi(\mathsf{N})$ could still be divisible by $2\mathsf{k}$. Some integers, even with unknown factoriza-

Q \ P								
k=0+4s	1	3	5	7	9	11	13	15
1	−1	−1	+1	+1	−1	−1	+1	+1
3	−1	−1	+1	+1	−1	−1	+1	+1
5	+1	+1	−1	−1	+1	+1	−1	−1
7	+1	+1	−1	−1	+1	+1	−1	−1
9	−1	−1	+1	+1	−1	−1	+1	+1
11	−1	−1	+1	+1	−1	−1	+1	+1
13	+1	+1	−1	−1	+1	+1	−1	−1
15	+1	+1	−1	−1	+1	+1	−1	−1

Q \ P								
k=2+4s	1	3	5	7	9	11	13	15
1	−1	+1	+1	−1	−1	+1	+1	−1
3	+1	−1	−1	+1	+1	−1	−1	+1
5	+1	−1	−1	+1	+1	−1	−1	+1
7	−1	+1	+1	−1	−1	+1	+1	−1
9	−1	+1	+1	−1	−1	+1	+1	−1
11	+1	−1	−1	+1	+1	−1	−1	+1
13	+1	−1	−1	+1	+1	−1	−1	+1
15	−1	+1	+1	−1	−1	+1	+1	−1

Q \ P								
k=1+4s	1	3	5	7	9	11	13	15
*	−1	−1	−1	−1	−1	−1	−1	−1

Q \ P								
k=3+4s	1	3	5	7	9	11	13	15
1	−1	+1	+1	−1	−1	+1	+1	−1
3	+1	−1	+1	−1	+1	−1	+1	−1
5	−1	+1	+1	−1	−1	+1	+1	−1
7	+1	−1	+1	−1	+1	−1	+1	−1
9	−1	+1	+1	−1	−1	+1	+1	−1
11	+1	−1	+1	−1	+1	−1	+1	−1
13	−1	+1	+1	−1	−1	+1	+1	−1
15	+1	−1	+1	−1	+1	−1	+1	−1

Figure 5.1: The Jacobi-symbol $J_{PQ}(k)$ for different residues of P and Q modulo 16.

tion, allow to obtain more information about the divisors of $\varphi(N)$. These are integers of the form $N = PQ^{2e}$, since one of the two involved primes is a square, which is ignored by the Jacobi symbol. In this way, the Jacobi symbol leaks information about the other prime involved. If N has the form $N = PQ^{2e}$, then for the Jacobi symbol and a co-prime integer $k > 2$, $J_N(k) = J_{PQ^{2e}}(k) = J_P(k) \cdot J_Q(k)^{2e} = J_P(k)$.

Using this fact, the tables displayed in Figure 5.2 show the values the Jacobi symbol $J_N(k)$ must adopt such that $2k$ does not divide $\varphi(P)$.

Example: Suppose $N = 132380144208075017604487$1 and N is of the form $N = PQ^{2e}$, $e > 0$. Suppose one wants to test whether $k = 41$ divides $P - 1$. Since $k \equiv 1 \pmod 4$, the third table must be used. Thus, $J_N(41) = -1$. The table shows that whenever the Jacobi symbol of k is negative, k can not divide $P - 1$.

Q \ P								
k=0+4s	1	3	5	7	9	11	13	15
*	−1	−1	+1	+1	−1	−1	+1	+1

Q \ P								
k=2+4s	1	3	5	7	9	11	13	15
*	−1	+1	+1	−1	−1	+1	+1	−1

Q \ P								
k=1+4s	1	3	5	7	9	11	13	15
*	−1	−1	−1	−1	−1	−1	−1	−1

Q \ P								
k=3+4s	1	3	5	7	9	11	13	15
*	−1	+1	−1	+1	−1	+1	−1	+1

Figure 5.2: The Jacobi-symbol $J_{PQ^{2e}}(k)$ for different residues of P and Q modulo 16.

In the next section, the last two tables are used to invalidate the Φ-Hiding assumption when using moduli of the form $N = PQ^{2e}$ and choosing P to hide the prime number in question.

5.3.3 Application to the Φ-Hiding Assumption

In both Definitions 5.2.1 and 5.2.2, it is only required that N is a composite integer with unknown factorization. By applying the results from the previous sections, it will be shown that this requirement is not sufficient. If the PHA is applied to a modulus of the form PQ^{2e}, where the integer P is constructed in such a way that P hides a given prime, then the Φ-Hiding assumption is violated with non-negligible probability. Moduli of this form, mostly with $e = 1$, are used by several cryptographic protocols, as described by Boneh and Shacham [22] and used, e.g., by Poupard and Stern [101], to speed up some computations that profit from the form PQ^{2e} with $e > 0$ instead of PQ. Using the results of the previous sections, the following theorem can be stated:

Theorem 5.3.9 *Let $N = PQ^{2e}$ and suppose that P hides p. Then, the Φ-Hiding assumption from Definition 5.2.2 can be violated. An attacker can choose the hidden prime with an advantage of*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{PHA}} = \left|\Pr[\mathcal{A}(N, p_1, p_2) = b] - \frac{1}{2}\right| = \frac{1}{4},$$

*which is **non-negligible**.*

The following notation is used: N is again of the form $N = PQ^2$ and $T(N, k)$ is the value of the corresponding table entry of Figure 5.2.

Proof 5.3.10 (of Theorem 5.3.9) *Suppose that either p_1 or p_2 divides $\varphi(N)$ and an attacker has to decide which of them divides $\varphi(N)$. Without loss of generality, we assume that p_1 is the prime that is hidden by P. For this prime, $J_N(p_1) \neq T(N, p_1)$ holds, because it divides $P - 1$ (see Theorem 5.3.6). Thus, the attacker will find at least one matching Jacobi symbol concerning the primes p_1 and p_2. From the attackers point of view, the probability that a prime p_i, $i \in \{1, 2\}$ divides $\varphi(N)$ is*

$$\Pr[\varphi(N) \equiv 0 \pmod{p_i}] = \begin{cases} 0, & J_N(p_i) = T(N, p_i) \\ 1, & J_N(\overline{p}_i) = T(N, \overline{p}_i) \\ \frac{1}{2}, & J_N(p_i) = J_N(\overline{p}_i) \end{cases} \tag{5.8}$$

where $\overline{p}_i$ denotes the other one of the two primes. Note the factorization of N is not needed to construct the tables in Figure 5.2. They are universally valid for moduli of the form $N = PQ^{2e}$ and thus known to the attacker. Whenever the Jacobi symbol $J_N(p_i)$ is equal to $T(N, p_i)$, Theorem 5.3.6 states that p_i cannot be a divisor of $\varphi(N)$, thus the probability is $\Pr[\varphi(N) \equiv 0 \pmod{p_i}] = 0$. Consequently, the Jacobi symbol $J_N(\overline{p}_i)$ must be not equal to $T(N, \overline{p}_i)$, which indicates that it is the hidden prime. If both Jacobi symbols do not match the table entry, no information is leaked and the attacker cannot argue in any direction. Thus, in this case the probability is $\Pr[\varphi(N) \equiv 0 \pmod{p_i}] = \frac{1}{2}$. Since the primes p_i are chosen randomly, it can be assumed that the Jacobi symbol $J_N(p_2)$ adopts random values of -1 and $+1$. The calculation of the total probability for the attacker to choose the hidden prime correctly is as follows: Whenever a Jacobi symbol evaluates to a value unequal to the table entry, it cannot be the prime that is hidden by P, so the attacker chooses the other one, the hidden one, with a probability of 1. When both Jacobi symbols evaluate to $\neq T(N, \cdot)$, the attacker chooses the right one with a probability of $\frac{1}{2}$. Thus, in total there is an average probability of $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$ to choose the correct prime, which proves Theorem 5.3.9. □

Composite Integers. The situation is even worse when the Φ-Hiding assumption is used with composite integers n_1 and n_2 instead of the primes p_1 and p_2, as done, for example, by Gentry et al. [49]. Assume that there is a modulus of the form $N = PQ^2$ and one wants to determine whether the composite integer n_i, which is the product of m distinct primes greater than 2, divides $\varphi(N)$. Suppose the Jacobi symbol is applied and the result does not allow to decide whether n_i divides $\varphi(N)$ or not. In this case, it can be proceeded with the prime factors of n_i. Since n_i is $\prod_{j=1}^{m} p_j$, the Jacobi symbol can simply be evaluated for all of its prime factors. If there is a prime p_j with a Jacobi symbol that leaks the required information, it can be concluded that n_i cannot divide $\varphi(N)$, since from $n_i | \varphi(N)$ it follows that $p_j | \varphi(N)$ must also hold. If the integers in question consist only of 7 prime numbers, there already is a success probability of $\approx 99\%$ to choose the right integer.

Corollary 5.3.11 *If $n_1 = \prod_{j=1}^{l_1} p_j$ and $n_2 = \prod_{j=1}^{l_2} q_j$ are two random, composite integers that are odd and square free Let $N = PQ^{2e}$ and suppose that P hides n_1. Then, the Φ-Hiding assumption from Definition 5.2.2 can be violated. An attacker can choose the hidden integer with an advantage of*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathsf{PHA}} = \left| \Pr[\mathcal{A}(N, n_1, n_2) = b] - \frac{1}{2} \right| = \frac{1}{2} - \frac{1}{2^{l_2+1}},$$

which is ***non-negligible****.*

$l_1 = l_2$	1	2	3	4	5	6	7
	0.5	0.75	0.875	0.938	0.969	0.984	0.992

Table 5.1: Success Probability

Proof 5.3.12 *Let $n_1 = \prod_{j=1}^{l_1} p_j$ and $n_2 = \prod_{j=1}^{l_2} q_j$ be two odd, square free integers. If $N = PQ^{2e}$ and exactly one of the two integers n_1 and n_2 divides $\varphi(N)$, the probability to choose the right one of the two possibilities is as follows. The case $l_1 = l_2 = 1$ was already addressed in the paper; it has a success probability of $\frac{3}{4}$. Note that if $n_i | \varphi(N)$, then also each divisor of n_i is a divisor of N. Thus, if we find a divisor of n_i that does not divide $\varphi(N)$, we can conclude that n_i is not the integer hidden by $\varphi(N)$. Since the same argument applies to all divisors that are prime numbers, it is sufficient to check all prime factors of n_i whether they are divisors of $\varphi(N)$ or not.*
Without loss of generality, we assume that n_1 is the integer hidden by $\varphi(N)$. For each of its l_1 prime factors p_i, $J_N(p_i) \neq T(N, p_i)$ must hold. For the other integer n_2, it follows that for each of its l_2 prime factors q_i it holds with a probability of $\frac{1}{2}$ that $J_N(q_i) \neq T(N, q_i)$ and with a probability of $\frac{1}{2}$ that $J_N(q_i) = T(N, q_i)$. Whenever the first case occurs, no knowledge is gained. But whenever the latter case occurs, the information that n_2 cannot be a divisor of $\varphi(N)$ is gained, so n_1 is the hidden number. The method fails if for all prime factors $J_N(q_i) \neq T(N, q_i)$ is obtained, which occurs with a probability of $\prod_{i=1}^{l_2} \mathsf{Prob}[J_N(q_i) \neq T(N, q_i)] = \frac{1}{2^{l_2}}$. Thus, the success probability of choosing the right integer is $(1 - \frac{1}{2^{l_2}})$. □

Table 5.1 illustrates the success probability of choosing the right prime for different numbers of prime factors.

5.4 Secret Sharing Schemes for $\varphi(N)$

Secret Sharing Schemes allow to distribute a secret among a set of users. Each user receives part of the secret from a trusted dealer. The entire secret can only be reconstructed if all participating users collaborate. If already a subset of users is sufficient to reveal the secret, the scheme is called a *Threshold Secret Sharing* scheme. More precisely, a scheme that allows t out of n users to reconstruct a secret, but not $t-1$ or less, is called a (n, t) threshold scheme.

The first practical secret sharing schemes were invented by Shamir [115] and Blakley [13], both in 1979. Shamir proposed to use a polynomial of degree $t-1$, say $f(x) =$

$s+\sum_{j=1}^{t-1} a_j x^j$, to share a secret s. In this case, the private part received from the dealer is a function point $(x, f(x))$ for a random value x. Only if at least t users collaborate, the function f can be reconstructed completely, e.g. by using the Lagrange interpolation method. After reconstruction, the computation of $f(0) = s$ finally reveals the secret to all participating users. The approach of Blakley is based on the intersection of n-dimensional hyperplanes. Whenever n n-dimensional and non-parallel hyperplanes intersect, they define a single point that is the hidden secret. Other secret sharing systems are based on the Chinese Remainder Theorem: the secret sharing scheme proposed by Mignotte [86] and the secret sharing scheme proposed by Asmuth and Bloom [6]. In both schemes, the secret integer s or a derivation of it is reduced modulo several co-prime integers. The emerging residues are the partial secrets distributed to each participating user. According to the definition of the CRT, these residues are sufficient to reconstruct the entire secret if a sufficient number of users collaborate.

The problem of any secret sharing system is that once the secret has been revealed, i.e. t or more users have decided to collaborate, the partial secrets are revealed, even those of the users that did not participate in the collaboration. Thus, the system must be reset and the dealer has to distribute new partial secrets. In some situations, this is quite inefficient, e.g. when the shared secret is a signing key and the system collapses each time a single signature is issued. Secret sharing schemes that overcome this problem are called *Function Sharing Schemes*. These schemes allow sharing a function, e.g. a signing function, among a set of users. The secret is shared among the users using a standard secret sharing scheme. Using the partial secret as the input for the shared function makes it inaccessible for others and allows users to conjointly create a signature without having to reveal their partial secret.

When the secret shared among a set of users using a secret sharing scheme is a truly random object, e.g. a string generated by a secure random number generator, combined parts of the secret should not reveal any information about its missing components. However, if the secret satisfies certain properties, this property cannot be guaranteed even if less than the required t users collaborate. For example, this is the case when the shared secret is the private integer d of the Rivest-Shamir-Adleman (RSA) [102] encryption system, satisfying the equation $ed = 1+\varphi(N)k$, where $\varphi(\cdot)$ is Euler's totient function, i.e. the number of positive integers less than or equal to N that are coprime to N. It has been shown that even partial information about the integer d is sufficient to reconstruct the entire integer d in polynomial time. These attacks are called *partial key exposure attacks* [45, 18, 14] and are based on the leakage of the most or least significant bits of the private integer d that can be obtained, for example, by side-channel attacks.

The leaked bits allow an adversary to generate an approximation of the actual integer d that gets more precise when more bits are leaked. Under certain conditions, the approximation and the public knowledge of the equation $ed = 1 + \varphi(N)k$ are sufficient to reconstruct the entire integer d. In a secret sharing scheme, each partial secret of a user can be viewed as an approximation of the secret. If users start to collaborate and start to combine their partial secrets, they get a better approximation that gets more precise as more users collaborate.

Boneh and Franklin [22] proposed to use a threshold-based approach to forfeit the abilities of an ID-PKG. If the ID-PKGs master keys are distributed across several ID-PKGs, no single instance gets into the knowledge of a users identity key. This clearly eliminates the drawback that someone else knows a private key, however it creates additional overhead and the need for new infrastructure. If their approach is applied to the SSF system, the integer $\varphi(N)$ is the secret that needs to be distributed.

In this section, it is demonstrated that at a certain point the approximation by malicious ID-PKGs is sufficient to recover the entire secret, which contradicts the definition of a secure (n, t) threshold secret sharing scheme. To the best of our knowledge, partial key exposure attacks against threshold sharing schemes have not been studied in the literature yet. The main contribution is to show that if the secret sharing scheme of Mignotte is used to share the secret key d of an RSA encryption system, as proposed by Iftene and Grindei [65], the secret can be revealed in polynomial time even with less than t users. An adversary who controls h users ($h < t$) can reconstruct the entire secret under the condition that the term $(t - h)/t$ is smaller than an upper bound that only depends on the size of d. For this purpose, the lattice-based reduction results, obtained by the analysis of partial key exposure attacks [45], are used. Furthermore, it is shown that the original definition of the secret sharing scheme of Asmuth and Bloom does not necessarily lead to a secure system. In particular, it is demonstrated that two of the three systems proposed by Kaya and Selcuk [67, 68] are insecure if an involved random integer is not sufficiently large. Additionally, it is shown that the secret sharing scheme of Asmuth and Bloom is not further vulnerable to lattice-based reduction attacks.

5.5 The Secret Sharing Scheme of Asmuth and Bloom

The secret sharing scheme proposed by Asmuth and Bloom is based on the Chinese Remainder Theorem. Informally, it utilizes the randomness that occurs if a random integer, say w, is reduced modulo certain integers m_i ($1 \leqslant i \leqslant n$). The generated residues are the partial secrets for the participating users. The CRT guarantees that if all m_i are pairwise co-prime, the integer w can be reconstructed from the residues uniquely. The following definition shows the steps executed during the Asmuth and Bloom sharing phase.

Definition 5.5.1 (Sharing in the Asmuth-Bloom scheme) *To share a secret s among a set of n participants in the secret sharing scheme of Asmuth and Bloom, the dealer does the following:*

1. *He choses a set of $n+1$ pairwise relative prime integers $m_0 < m_1 < ... < m_n$ with $M = \prod_{i=1}^{t} m_i$ and:*

$$M > m_0 \prod_{i=1}^{t-1} m_{n-i-1} \tag{5.9}$$

2. *He computes w with $0 \leqslant w = s + m_0 \cdot A < M$, where A is chosen randomly from $\mathbb{N}$.*

3. *He computes the part of the secret of the i-th user by $w_i \equiv w \pmod{m_i}$*

Definition 5.5.1 is the original definition of the Asmuth and Bloom sharing phase. In this form, it is used in several protocols. Next, the definition for the reconstruction of the secret is presented.

Definition 5.5.2 (Reconstruction in the Asmuth-Bloom scheme) *Let $\mathcal{S}$ be a set of t collaborating users and let $M_{t,\mathcal{S}} = \prod_{i \in \mathcal{S}} m_i$ be the product of the corresponding modulo values. Furthermore, $I_i \frac{M_{t,\mathcal{S}}}{m_i} \equiv 1 \pmod{m_i}, i \in \mathcal{S}$. To reconstruct a secret s in the secret sharing scheme of Asmuth and Bloom, each user computes:*

1. $u_i \equiv w_i I_i \frac{M_{t,\mathcal{S}}}{m_i} \pmod{M_{t,\mathcal{S}}}$

The trusted combiner collects all values u_i and computes:

1. $w = \sum_{i \in \mathcal{S}} u_i \pmod{M_{t,\mathcal{S}}}$

2. $s \equiv w \pmod{m_0}$

The reconstruction can also be invoked with less than t users, which yields an approximation of the secret s. Such an approximation can always be used to write

$$w = \hat{w} + M_{h,\mathcal{S}} v \tag{5.10}$$

where $\hat{w}$ is the approximation, $M_{h,\mathcal{S}}$ is the product of the moduli values of the collaborating users and v is some unknown integer that is smaller, the more users collaborate ($v = 0$, if $h = t$). The equation follows directly from the reconstruction definition of the CRT.

The next sections contain attacks to the secret sharing scheme of Asmuth and Bloom as defined above: (a) concerning the small random integer A (see Definition 5.5.1, Step 2), and (b) using lattice-based reduction.

5.5.1 Implications of the Small Random Integer A

Several protocols were invented during the last years that use secret sharing schemes for different purposes. Recently, Kaya and Selcuk [67, 68], have proposed three robust function sharing schemes that use the secret sharing scheme of Asmuth and Bloom to distribute the partial secrets. The first function sharing scheme is a robust signature system based on the RSA algorithm. The second and third function sharing schemes are extension of their ideas to the Paillier encryption system [94] and the ElGamal encryption system [43], respectively. In the next section, it is shown that Definition 5.5.1 of the secret sharing scheme of Asmuth and Bloom, also used by Kaya and Selcuk, leads to an insecure function sharing scheme. The three function sharing schemes proposed in these two papers are discussed and the weaknesses for two of them are demonstrated. No lattice-base reduction methods are used for this purpose, but it is shown that a straightforward computation can factor the entire modulus in the RSA and Paillier cases if the integer A is too small.

The problem that can occur when A is too small is that not only $w \leqslant M$ holds, but it is also possible that $w < M_{h,\mathcal{S}}$, where $M_{h,\mathcal{S}}$ is the product generated by the m values hold by the attacker. Thus, the integer v in Equation 5.10 is zero. In this case, the attacker obtains the secret w by simply invoking the reconstruction algorithm with h users. The adversary can now use the integer w not only to break the function sharing scheme, but also to factor the used modulus. For the rest of the section it is assumed that whenever an RSA integer $N = pq$ is involved, it is a balanced RSA integer, which means that both prime factors are of equal size. This property is one of the mandatory features that makes an RSA integer more difficult to factor.

5.5.1.1 Threshold RSA Signatures

During the setup of the function sharing schemes proposed by Kaya and Selcuk [67, 68], the authors suggest to set $m_0 = \varphi(N)$. Despite the original definition of the secret sharing scheme of Asmuth and Bloom, they also require that m_0 is kept secret, *"to prevent the participating users to factor the public modulus N"*. The secret integer w is constructed by $w = d + \varphi(N)A$ and A is chosen such that $0 \leqslant w < M$. y is now a combination of three integers, all unknown to the participating users. If A is not sufficiently large, such that $w < M_{h,S}$, an adversary can recover w. However, he cannot use w directly to recover d or $\varphi(N)$. But since the integer d is part of the known equation $ed = 1 + \varphi(N)k$, the adversary can use the recovered w and transform the contained equation into the following equation:

$$W = ew - 1 = ed - 1 + e\varphi(N)A = \varphi(N)(eA + k) \tag{5.11}$$

Thus, after a multiplication with the RSA public integer e and a subtraction of 1, the adversary obtains an integer W that is a multiple of $\varphi(N)$. This integer W can now be used to recover the factorization of N in probabilistic polynomial time using the following well known algorithm:

Algorithm 23 Factoring using a multiple of $\varphi(N)$

```
Input: $\varphi(N)A$, $N$
Output: a factor of $N$
1. $var \leftarrow \varphi(N)A$
2. while $var$ is even
3.     $var \leftarrow var/2$
4. $R \overset{rand}{\leftarrow} \mathbb{Z}_N$
5. $r \leftarrow R^{var} \pmod{N}$
6. if $(N > g_1 = \gcd(r+1, N) > 1)$ or $(N > g_2 = \gcd(r-1, N) > 1)$
7.     return $max(g_1, g_2)$
8. else goto 4.
```

The algorithm succeeds roughly with a probability of $1 - \frac{1}{2^l}$ after l trials. Obviously, if w can already be reconstructed by $t-1$ users, they can also already issue valid signatures.

However, they are only able to reveal the partial secrets of the other (honest) users if they factor the modulus. This can be done as described above, even if m_0 is kept secret to the dealer.

5.5.1.2 Threshold Paillier Signatures

Kaya and Selcuk have extended their ideas to Paillier's encryption system [94]. Paillier's encryption system is also based on the integer factorization problem, but uses the trapdoor to compute discrete logarithms in $\mathbb{Z}^*_{N^2}$. The system makes use of the λ function, defined as $\lambda(N) = \mathrm{lcm}(p-1, q-1)$, if $N = pq$, which makes $\lambda(N)$ always less than N. In this case, the authors construct the secret integer as $w = \lambda + \varphi(N^2)A = \lambda + N\varphi(N)A$. If the adversary obtains w because of a small chosen A and since $w < N < M_S$ and N is public, the adversary can simply compute $\lambda(N) = w \pmod N$ and thus obtains $\lambda(N)$, which can be used to factor N using Algorithm 23.

5.5.1.3 Threshold ElGamal Encryption

The third proposal of Kaya and Selcuk is an extension to the ElGamal encryption system [43]. The ElGamal system is an encryption system based on the discrete logarithm problem and is usually defined in $\mathbb{F}_p$, for a suitable prime number p. In this case, using the integer $\varphi(p)$ as the dealer's private m_0 is useless, since p is a prime number and thus $\varphi(p)$ is equal to $p-1$. As a workaround, the authors propose to again use a composite modulus N, with factors $p = 2p' + 1$, $q = 2q' + 1$ and $p, q, p', q' \in \mathbb{P}$. The secret y is now constructed by $w = \alpha + 2p'q'A$, where $g^\alpha \equiv \beta \pmod{pq}$, with (pq, β, g) public. In this case, α, $2p'q'$ and A are unknown. No simple property, like the RSA equation, can be used here. Despite the fact that the collaborating users can issue signatures, there is no method to factor the modulus N in this case.

5.5.2 Lattice-Based Reduction and the Asmuth-Bloom Secret Sharing Scheme

The main application area for lattice-based reduction in cryptography is to find roots of polynomials. These roots can be, for example, the integer $p+q$ of an RSA modulus or even the plain text of an encryption scheme. The requirement to find these roots is that they are smaller than a certain upper bound that can be extracted out of the lattice that is constructed using the coefficient vectors of the polynomial in question. To get any new information from the constructed lattice, it has to be reduced to another lattice with shorter vectors. Even if the challenge to find the shortest vector in a lattice is NP-hard, a sufficiently short vector is often enough to find the roots in question. Such short vectors can be found, for example, with the LLL-Algorithm [73] that runs in polynomial time. The proper construction of a lattice is the key point for success, so often not only the original polynomial is used for the construction, but even several carefully chosen alterations of it. For more details on the application of lattice theory in cryptography, the reader is referred to the papers of Coppersmith [37] and Coron [38].

In the previous section, it was shown that if $w < M_{h,\mathcal{S}}$, even a set of $t-1$ (or even less if A is sufficiently small) users can recover the entire secret. The reason is that the original definition of the secret sharing scheme of Asmuth and Bloom does not require the secret to be in a secure range (like Mignotte's secret sharing scheme does), but only to be smaller than M. In this section, it is assumed that this flaw is corrected and $w = d + \varphi(N)A$ is made sufficiently large by choosing an appropriate value for A. Note that also here A cannot be arbitrarily large, since nevertheless $w = d + \varphi(N)A < M$ must hold, such that a set of t users can recover w in a unique way. Next, it is shown that Asmuth and Bloom's secret sharing scheme, in contrast to the secret sharing scheme of Mignotte, is in this case almost secure against lattice-based reduction attacks based on the approximation of w, despite the values $t = 3$ and $t = 4$ when an adversary controls a set of $t-1$ users ($M_{h,\mathcal{S}} = M_{t-1,\mathcal{S}}$) and thus gets the best possible approximation of w.

After the secret w has been shifted into a sufficient size by means of choosing a larger value for A, w is bounded by the following integers:

$$M > w = d + \varphi(N)A > M_{t-1,\mathcal{S}} \tag{5.12}$$

Since w is now larger than $M_{t-1,\mathcal{S}}$, it cannot be completely revealed by less than t users, but it can be approximated. This approximation can be written as $w = \hat{w} + M_{t-1,\mathcal{S}}v$. Again, an adversary can utilize the relationship $ed = 1 + \varphi(N)k$, which enables him or her to rewrite the approximation equation as (using $\varphi(N) = N - (p + q - 1)$)

$$ew = 1 + N(eA + k) - (p + q - 1)(eA + k) = e\hat{w} + eM_{t-1,\mathcal{S}}v \tag{5.13}$$

To simplify the equation, $\hat{A} = eA + k$ as well as $R = e\hat{y} - 1$ is used. Furthermore, it is $x_0 = v, y_0 = \hat{A}, z_0 = p + q - 1$, which finally yields a function f of three unknowns

$$f(x, y, z) = eM_{t-1,\mathcal{S}}x - Ny + zy + R \tag{5.14}$$

with a root at $f(x_0, y_0, z_0)$. The arising monomials of the obtained function $f(x, y, z)$ match exactly those of the f_{LSB} function defined by Ernst et al. [45]. Thus, their shift-polynomials as well as their lattice can be used. Even though the monomials are equal, the proportion among the variables differs in our case. Therefore, it is necessary to define new upper bounds X, Y and Z for the three corresponding variables x,y and z.

The upper bound of x. From the inequality $M > w = d + \varphi(N)A = \hat{w} + M_{t-1,\mathcal{S}}x_0$ it can be deduced that $m_t > \frac{M}{M_{t-1,\mathcal{S}}} > \frac{(M-w)}{M_{t-1,\mathcal{S}}} > x_0$. Since from the basic requirement of the secret sharing scheme of Asmuth and Bloom that $\prod_{i=1}^{t} m_i > m_0 \prod_{i=1}^{t-1} m_{n-i-1}$, the m_i values cannot be too different in size. This allows estimating the size of m_t with $x < m_t < M_{t-1,\mathcal{S}}^{1/(t-1)+\epsilon} = X$, for some small value ϵ that depends on the gaps between m_i.

The upper/lower bound of y. For the upper bound Y for $\hat{A}$, we just write $Y = M_{t-1,\mathcal{S}}^{\beta} > A$ and leave β unspecified in terms of t. Next, a lower bound for β is approximated. The requirement is that $m_n M_{t-1,\mathcal{S}} > d + m_0 A > M_{t-1,\mathcal{S}}$ holds. Dividing by m_0 yields

$$\frac{m_n M_{t-1,\mathcal{S}}}{m_0} > \frac{d}{m_0} + A > \frac{M_{t-1,\mathcal{S}}}{m_0} \tag{5.15}$$

and since $d < m_0 = \varphi(N)$, it holds $A > \frac{M_{t-1,\mathcal{S}}}{m_0} - 1 > M_{t-1,\mathcal{S}}^{(t-2)/(t-1)} - 1$, thus a lower bound is obtained with $\beta > (t-2)/(t-1)$.

The upper bound of z. The integer z_0 is equal to $z = N - \varphi(N) = p + q - 1$. Since it is assumed that N is a balanced RSA integer, $z < 3\sqrt{N} < 3\sqrt{2\varphi(N)} < 5\sqrt{\varphi(N)}$. Since the integer m_0 is per definition (see Definition 5.5.1) less than any m_i involved in the product $M_{t-1,\mathcal{S}}$, it follows that $z < 5M_{t-1,\mathcal{S}}^{1/(2(t-1))} = Z$.

The three upper bounds can be substituted into the determinant inequality [45] which yields

$$X^{1+3\tau}Y^{2+3\tau}Z^{1+3\tau+3\tau^2} \leqslant W^{1+3\tau} \tag{5.16}$$

with $W = \max\{eM_{t-1,\mathcal{S}}X, NY, YZ, R\} \geqslant eM_{t-1,\mathcal{S}}X$. After substituting the calculated bounds, this turns into

$$M_{t-1,\mathcal{S}}^{(1+3\tau)/(t-1)} M_{t-1,\mathcal{S}}^{\beta(2+3\tau)} M_{t-1,\mathcal{S}}^{(1+3\tau+3\tau^2)/(2(t-1))} \leqslant M_{t-1,\mathcal{S}}^{(1+2/(t-1))(1+3\tau)} \tag{5.17}$$

Using Ernst et al.'s [45] optimal value for $\tau = \frac{1}{2} - \delta = \frac{1}{2} - \frac{1}{t-1}$ and solving the equation to β yields a dependency of β and the number of malicious users (in this section fixed to $t-1$):

$$\beta \leqslant \frac{20t^3 - 77t^2 + 94t - 49}{28t^3 - 108t^2 + 132t - 52} \tag{5.18}$$

During the computation of the upper bound for y, it was also obtained a lower bound for β depending on t: $\beta > (t-2)/(t-1)$. Plugging this into the equation above yields

$$0 < \beta - \frac{t-2}{t-1} \leqslant \frac{-8t^3 + 59t^2 - 118t + 55}{28t^3 - 108t^2 + 132t - 52} \tag{5.19}$$

which is fulfilled only for $t = 3$ and $t = 4$. In the case $t = 3$, β has to be between $5/8 = 0.625$ and 0.5, and in the case $t = 4$, β has to be between $25/36 = 0.69\overline{4}$ and $2/3$, which are both possible combinations.

In this section, attacks against the secret sharing scheme of Asmuth and Bloom were presented. The first attacks based on the small value A make a system using this parameter insecure. However, this attack can easily be circumvented by choosing A of sufficient size. The lattice-based reduction attacks are also of limited effect with respect to the secret sharing scheme of Asmuth and Bloom, since the secret equation w prevents the attacker from getting a better approximation of d.

5.6 The Secret Sharing Scheme of Mignotte

In this section, it is shown that Mignotte's secret sharing scheme is much more vulnerable against lattice-based reduction attacks and can be attacked whenever the number of malicious users exceeds a bound that depends on the size of d. This difference between Asmuth and Bloom's secret sharing scheme and Mignotte's secret sharing scheme occurs in the sharing phase where the size of the involved integers as well as the partial secret computation differs.

Definition 5.6.1 (Sharing in Mignotte's scheme) *To share a secret s among a set of n users in Mignotte's secret sharing scheme, the dealer does the following:*

1. *He choses a set of $n+1$ pairwise relative prime integers $m_0 < m_1 < ... < m_n$ with $M = \prod_{i=1}^{t} m_i$ and:*
$$M > s > \prod_{i=1}^{t-1} m_{n-i-1} \tag{5.20}$$

2. *He computes the part of the secret of the i-th user by $s_i \equiv s \pmod{m_i}$*

In the sharing phase, it is required that s lies in a secure interval. This requirement prevents an adversary to apply the attacks presented in Section 5.5.1. No subset of less than t users is able to generate the secret integer s. However, in contrast to the secret sharing scheme of Asmuth and Bloom where the secret s in embedded into an equation, the i-th user gets a partial secret s_i that is the direct remainder of $s \pmod{m_i}$. This allows the successful application of the lattice-based reduction method, which fails in the secret sharing scheme of Asmuth and Bloom, since the obtained approximation of the colluding users is more precise in this case.

In the sequel, it is assumed that an adversary who controls h users with $1 \leqslant h < t$ exists. The following theorem is proved:

Theorem 5.6.2 *Let d be the private integer of a balanced-RSA encryption system with $d < N^{\beta}$. Suppose d is shared with Mignotte's secret sharing scheme with n users and threshold t. If an adversary controls h users such that $(t-h)/t < \frac{1}{6}(5 - 2\sqrt{1+6\beta})$ holds, the adversary can recover d in polynomial time.*

Proof 5.6.3 *The approximation generated by the h collaborating users can be written as $d = d_0 + M_{h,\mathcal{S}} d_1$, with $d_1 \in \mathbb{N}$ unknown as long as $s < t$. Again, the adversary can utilize the publicly known equation $ed = 1 + \varphi(N)k$ to insert the approximation. The result is Equation (5.21):*

$$eM_{h,\mathcal{S}} d_1 + ed_0 - 1 - Nk + (p+q-1)k = 0 \tag{5.21}$$

The monomials equal those of the previous case, but this time with N being the dominant factor. After the unknowns are renamed to: $x_0 = d_1, y_0 = k, z_0 = p+q-1$, a linear polynomial (with $R = ed_0 - 1$)

$$f(x,y,z) = eM_{h,\mathcal{S}} x + R - Ny + zy \tag{5.22}$$

is obtained that has a root at $f(x_0, y_0, z_0)$.

In this case, the bounds X,Y *and* Z *are defined according to the integer* N. *The number of involved malicious users is not considered yet. We simply set* $x < X = N^{\delta}$, $y < Y = N^{\beta}$, $z < Z = 3N^{1/2}$, *and use again the equation*

$$X^{1+3\tau}Y^{2+3\tau}Z^{1+3\tau+3\tau^2} \leqslant W^{1+3\tau} \tag{5.23}$$

with $W = \max\{eM_{h,\mathcal{S}}X, NY, ZY, R\} \geqslant eM_{h,\mathcal{S}}X = N^{1+\beta}$. *After inserting the bounds for* X, Y *and* Z, *this equation turns into*

$$N^{\delta(1+3\tau)}N^{\beta(2+3\tau)}N^{(1+3\tau+3\tau^2)/2} \leqslant N^{(1+\beta)(1+3\tau)} \tag{5.24}$$

Using Ernst et al.'s [45] optimal value for $\tau = \frac{1}{2} - \delta$, *the solution of the equation is*

$$\delta \leqslant \frac{1}{6}(5 - 2\sqrt{1+6\beta}) := B(\beta) \tag{5.25}$$

This is exactly the bound also obtained by Ernst et al.

What is missing is to reconstruct the number of malicious users sufficient to establish the attack. Thus, the values δ *or* β *have to be linked with the number of malicious users* h. *Per definition it was required that the secret* s, *in this case* d, *must lie between*

$$\prod_{i=1}^{t} m_i > d = d_0 + M_{h,\mathcal{S}}d_1 > \prod_{i=1}^{t-1} m_{n-i-1} \geqslant M_{h,\mathcal{S}} \tag{5.26}$$

To obtain a general formula, one can write $m_i \approx< d^{1/t+\epsilon_i}$ *for some small values* ϵ_i. *These values depend on the differences of the* m_i *integers. Since these integers cannot be too different in size because of the definition* $M > s > M_{t-1,\mathcal{S}}$, *these* ϵ_i *must be small. In the following, we let these* ϵ *values all contribute to some error term* $\mathcal{O}(\epsilon)$. *Now, the product over the* t m_i *values is larger than* d, *and a product with less factors is not smaller than* d. *Thus,* d_1 *has to be smaller than* $d^{(t-h)/t}$. *Otherwise,*

$$M_{h,\mathcal{S}} \cdot d_1 = \prod^{h} d^{1/t+\epsilon_i} \cdot d_1 > \left(\prod^{h} d^{1/t+\epsilon_i}\right) d^{(t-h)/t} = d^{1+\mathcal{O}(\epsilon)} \tag{5.27}$$

holds, which would contradict the requirement from Equation (5.26). This finally leads to

$$d_1 < N^{(t-h)/t} = N^{\delta} \quad \Leftrightarrow \quad (t-h)/t = \delta \tag{5.28}$$

β	t	h	$\frac{t-h}{t}$	$B(\beta)$
0.1	50	30	0.4	0.411
0.2	50	34	0.32	0.338
0.3	50	37	0.26	0.275
0.4	50	40	0.2	0.218
0.5	50	42	0.16	0.166
0.6	50	45	0.1	0.118
0.7	50	47	0.06	0.073
0.8	50	49	0.02	0.030

β	t	h	$\frac{t-h}{t}$	$B(\beta)$
0.1	20	12	0.4	0.411
0.2	20	14	0.3	0.338
0.3	20	15	0.25	0.275
0.4	20	16	0.2	0.218
0.5	20	17	0.15	0.166
0.6	20	18	0.1	0.118
0.7	20	19	0.05	0.073
0.8	20	**20**	**0**	0.030

Figure 5.3: Required number of malicious collaborators such that $\frac{t-h}{t} < \frac{1}{6}(5 - 2\sqrt{1+6\beta}) := B(\beta)$

which proves the theorem. ***Q.e.d.***

The integer t is a fixed parameter for the adversary he cannot change. However, the integer h is a variable that states how much partial secrets he must reveal to break the sharing scheme. It can be seen that whenever the adversary controls all necessary t users, thus $h = t$, the term $(t - h)/t$ is zero, making the inequality always true. This is necessary, since t users are per definition sufficient to reveal the secret.

Experimental Results

In this section, several experimental results with respect to attacking Mignotte's secret sharing scheme are presented. All measurements were performed on a Core2Duo 2.4 GHZ with 2 GB RAM under Windows XP using an implementation in Mathematica v4.1. To build the lattice, the same set of shift-polynomials as defined by Ernst et al. [45] are used. Since the performance of Mathematica is worse than optimized C code, the obtained results are not optimal in terms of speed, but nevertheless show the practicability of the attack.

Figure 5.3 shows the theoretical number of malicious users that are at least required to perform the attack. Note that the term $\frac{1}{6}(5 - 2\sqrt{1+6\beta})$ gets negative for $\beta > 7/8$, thus only the values for β up to 0.8 are taken. The left table shows the bounds for a threshold of a $(50, n)$ setup, whereas the right table is for $(20, n)$.

For very small values of $d = N^{\beta}$, it is evident that nearly only half of the intended users are sufficient to recover the entire secret. The more the secret integer grows, the more users are needed to break the scheme, which is consistent with the results from partial key exposure attacks. When a size of $\beta = 0.8$ is reached, only one of the $t = 50$ users can be left out, which nevertheless breaks the definition of a (n, t) secret sharing scheme threshold scheme. On the contrary, in the case $t = 20$ and $\beta = 0.8$, the attack fails, which is due to the smaller value of $t = 20$, rather than $t = 50$.

Since the bounds obtained by Ernst et al. [45] are only asymptotic, they are not always reached in practice. We made some measurements for a 256-bit modulus, showing what values of h can be achieved in practice. The dimension of the lattice is varied by changing the m and f values of the shift-polynomials:

$$\begin{aligned}
&g_{ijk}(x,y,z) = x^i y^j z^k F(x,y,z) X^{m-i} Y^{m-j} Z^{m+f-k},\\
&\text{for } i = 0,\dots,m,\ j = 0,\dots,m-i,\ k = 0,\dots,j\\
&h_{ijk}(x,y,z) = x^i y^j z^k F(x,y,z) X^{m-i} Y^{m-j} Z^{m+f-k},\\
&\text{for } i = 0,\dots,m,\ j = 0,\dots,m-i,\ k = j+1,\dots,j+f\\
&g'_{ijk}(x,y,z) = N x^i y^j z^k,\\
&\text{for } i = 0,\dots,m+1,\ j = 0,\dots,m+1-i,\ k = 0,\dots,j\\
&h'_{ijk}(x,y,z) = N x^i y^j z^k,\\
&\text{for } i = 0,\dots,m+1,\ j = 0,\dots,m+1-i,\ k = j+1,\dots,j+f
\end{aligned}$$

where $F(x,y,z) = R^{-1} f(x,y,z) \pmod{N} \equiv 1 + ax + by + cyz$.

In Figure 5.4, some measurements are shown. Using a lattice with a dimension of 16, is was only possible to attack d values up to size $N^{0.4}$. Using a lattice with a dimension of 40, it was also possible to get results for $d = N^{0.6}$. For higher values, no solutions in a reasonable amount of time could be found, which is probably due to the Mathematica implementation. The runtime of the Mathematica method `LatticeReduce[]`, performing the LLL-Algorithm, is the major runtime component and its used time is shown in the last column of Figure 5.4 (in milliseconds). The solutions found are similar to the theoretical bounds and demonstrate the practicability of the presented attack. Note, if $\beta \leqslant 0.292$ the private key can already be obtained by attacks from Boneh and Durfee [17] or Wiener [127].

5.7 Summary

Φ-Hiding Assumption. In the first part of this chapter, it was shown that in some circumstances it can be efficiently decided whether a given prime p divides $\varphi(N)$ or not. This can be done despite the factorization of N is unknown and if N is of the form

256-bit modulus				
β	t	h	Dim	ms
0.1	20	7	16 (m=f=1)	2031
0.2	20	8	16 (m=f=1)	2131
0.3	20	10	16 (m=f=1)	4121
0.4	20	14	16 (m=f=1)	4422
0.5	20	16	30 (m=2 f=1)	63549
0.6	20	18	40 (m=2 f=2)	720312

Figure 5.4: Measurements using a 256-bit modulus.

$N = PQ^{2e}$, $e > 0$ and P hides the prime in question. The findings are based on the novel approach to utilize if a certain equation is defined over $\mathbb{Z}_N$, which can be tested by the Jacobi symbol. If someone implements a cryptographic protocol based on the Φ-Hiding assumption and uses such moduli, an attacker has an average probability of $\frac{3}{4}$ to choose the right prime, if the primes the attacker can choose from are selected randomly.

In cases when it is desired to ask which composite number n_i is hidden by P, the success probability would be even greater than $\frac{3}{4}$, since for each prime factor of n the attacker has the success probability of $\frac{3}{4}$.

There are two possible countermeasures to the presented attack. First, moduli of the form $PQ^{2e}, e > 1$ should not be used in conjunction with the Φ-Hiding assumption. Second, the primes a user can choose from should not be selected randomly, but only those primes that have a positive Jacobi symbol regarding N should be used. However, the assumption as stated in the original form must be corrected to exclude this cases where an attacker has non-negligible success probability.

Secret Sharing Schemes. The second part of this chapter was about an attack on CRT-based threshold secret sharing schemes in the case when they are utilized to share the integer $\varphi(N)$ among a set of users. Based on the generalized partial key exposure attacks by [45], it was proven that collaborating malicious users can break the scheme even if their number less than the required threshold. The combination of their partial secrets leads to an approximation which can be used, together with lattice-based reduction methods, to recover the entire secret in polynomial time.

6 Applications

”Lots of people working in cryptography have no deep concern with real application issues. They are trying to discover things clever enough to write papers about.”
Whitfield Diffie

6.1 Introduction

In this section, applications for the proposed protocol and the required issues for practicability are discussed. Identity-based cryptography has several applications. It can be applied to each environment where entities, human or artificial, have a unique identifier that is used for communication. Examples are e-mail communication, secure function or service calls in software architectures or GPS coordinates. However, here we focus on the following two areas:

- **IP networks**. One of the most obvious choices when considering the Internet is to choose an user’s Internet protocol address as its identity, hence as its public key. During the construction of a working architecture, several problems were identified:

  1. *How to distribute of the pubic shared parameters?*

  2. *How to distribute of the identity keys?*

  3. *How to handle dynamic IP addresses?*

  4. *How to handle key expiration and network address translation (NAT)?*

  Furthermore, the problem of address spoofing or IP spoofing in particular, must be solved. It will be demonstrated how the SSF signature scheme can be used to narrow down the ability to fake a source address of a packet by signing a certain timestamp.

- **Telephony**. The second choice is human communication. We distinguish between GSM and VoIP telephony. In the first case, the telephone number and in the second case, the VoIP address is used as the public key.

At the end of this chapter, an optimization of the proposed protocol is shown that works for other cryptographic protocols as well. This optimization is in the area of server-aided cryptography and is especially useful for the GSM scenario, where complex algorithms must be executed on less powerful hardware. By outsourcing costly algorithm to high-capacity servers, the speed for the actual online computations can be increased significantly.

The results of this chapter were published in [110], [122], [121], [106].

6.2 SSF in IP Networks

IP networks are a well-working scenario for identity-based cryptography since each participant has a unique identifier that is used for communication. Furthermore, the existing infrastructure already possesses a solution to lookup the identity of a foreign host, that is the *domain name service* (DNS). DNS translates a web-address to the actual IP address of the hosted server, which is equal to a public key distribution system when using IBC.

6.2.1 Distribution of Shared, Public Parameters

The distribution of public shared parameters is only necessary if more than one ID-PKG is available. Since the proposed scheme focuses heavily on this case, this must be considered. It should be noted that a main requirement is to try to minimize the number of global distribution steps in favor of local distribution steps, since this distributes the workload and reduces the risk of a global compromise. In a scenario with $\#\mathsf{prov}$ providers, each with $\#\mathsf{cust}$ customers where $\#\mathsf{cust} \gg \#\mathsf{prov}$, there are $\#\mathsf{prov} \cdot \#\mathsf{cust}$ customers in total. This means that $\#\mathsf{prov} \cdot \#\mathsf{cust}$ private/identity keys need to be distributed. In a PKI, in the worst case in which everybody wants to communicate with everybody else, $(\#\mathsf{prov} \cdot \#\mathsf{cust} - 1) \cdot (\#\mathsf{prov} \cdot \#\mathsf{cust})$ public keys need to be exchanged and managed. In SSF system, only the public parameters of the $\mathfrak{m}$ providers need to be exchanged. This reduces the number of transfers from $\#\mathsf{prov} \cdot \#\mathsf{cust}$ local and $(\#\mathsf{prov} \cdot \#\mathsf{cust} - 1) \cdot (\#\mathsf{prov} \cdot \#\mathsf{cust})$ global transfers to $\#\mathsf{prov} \cdot \#\mathsf{cust}$ local transfers and only $\#\mathsf{prov}$ global transfers, and since $\#\mathsf{cust} \gg \#\mathsf{prov}$, this is a large saving. Even using traditional key distribution mechanisms, SSF system offers a significant saving compared to a PKI in key escrow mode. In the following, further optimizations of the distribution process which are possible due to the network centric approach of the proposed solution will be suggested.

Like most other IBC approaches, the proposed system also uses shared public parameters. In a single domain scenario, the distribution of the public parameters is not a problem. However, if each AS runs its own ID-PKG, the number of public parameters and the binding between public parameters and identity keys becomes more complex. As stated above, this distribution problem is still much smaller than the distribution problem for traditional public keys where each entity has its own public key which needs to be distributed. Of course, traditional PKI technology can be used to distribute the public parameters, but a more suitable solution is to integrate the public parameters into the DNS lookup messages. In this way, the fact that a DNS lookup is made anyway to resolve a host IP is utilized, and the public parameter transfer can be piggybacked to the DNS reply. The technical details of the integration of IBE public parameter information into DNS records were evaluated by Smetters and Durfee [120]. The positive evaluation lead us to adopt the public parameter distribution technique for the SSF system. For more information on the details of how to incorporate this kind of information into the DNS system, the reader is referred to [120] or [1]. To secure the transport, either DNSsec can be used or the public parameters can be signed and transfered with standard DNS, or a key agreement can be executed between the requesting party and the DNS server if the public parameters of the DNS server are known. Since the DNS server is usually in the same AS as the requesting customer, this is not a problematic issue, because the public parameters are the same as the customer's public parameters. As stated above, this part of the system has been tried and validated by several research groups.

6.2.2 Distribution of the Identity Keys

The most critical element in all IBEs or PKIs in key escrow mode is the distribution of the identity keys (private keys) and the prevention of identity misbinding. In traditional PKI and IBE systems, this is usually done manually and out-of-band and thus creates a lot of work. While it can be argued that due to the fact that on the AS level most customers receive an out-of-band message when they receive their endpoint address, adding a fingerprint to the identity key would not put much extra burden on the system. However, a far more elegant solution for the long term is to integrate the key distribution into the IP distribution system. For most networks, this means integration into the DHCP server. This, however, is not trivial since DHCP on its own is an unsecured protocol not suitable for transferring private information. The two main threats are packet sniffing and MAC spoofing. If the identity key is sent in the clear via the DHCP protocol in an unswitched network, an attacker can sniff the identity key, leading to key compromise. With MAC spoofing, an attacker pretends to be the legitimate owner of a foreign MAC address, and the DHCP server sends the identity key to the attacker. Both forms of attacks make the plain use of DHCP for key distribution infeasible. In the following, several solutions are presented geared towards different scenarios of how the distribution of identity keys can be integrated into DCHP securely. In a fixed corporate network environment using a switched infrastructure, the easiest solution is to use the MAC lockdown function of modern switches. Using MAC lockdown, each port gets a MAC address and will only serve that MAC address. Thus, if an attacker wishes to spoof a MAC address to gain the key, physical access to the correct port must be acquired, significantly increasing the risk and effort of the attack. This scenario works fine in a corporate network where each MAC address is registered and assigned to a port anyway. In a student dormitory, for example, it is less feasible since managing the ever changing MAC addresses of the private devices used by students would be very time consuming and error prone. Here, an IEEE 802.1X + Radius [36] solution is more practical. The authorization is usually done in the form of a user-name password check. The IP address and the corresponding identity key can either be fixed (as set by the Radius and DHCP server) or dynamic and transient. Either way, only the legitimate user receives the identity key, and it is not possible to spoof the MAC address to receive a copy in the same key lifetime. If packet sniffing is an issue, the DHCP request needs to be extended to include a protected session key with which the identity can be protected from sniffing attacks. The client creates a session key which is encrypted using the public parameter N (N can be used in the same way as an RSA public key) of the key generator of the DCHP server and broadcasts the

DHCP request. The session key can only be decrypted by the DHCP server who then uses the session key to encrypt the identity key of the client, using e.g. the Advanced Encryption Standard AES, which is then broadcasted. Thus, the identity key can only be decrypted by the client. Apart from these two practical solutions based on an extension of existing security mechanisms which can be used in the short term, it is also presented a more speculative long term solution which does not rely on other security mechanisms. In this case, the network layer key agreement scheme is bootstrapped on the data link layer by using MAC addresses as public keys. As with IP addresses, it cannot be assumed that there will be a single authority to generate the MAC identity keys, but since the proposed system does not require cooperation between the ID-PKGs, this can be handled. Each organization with the authority to distribute MAC addresses runs its own ID-PKG and writes the identity key onto the networking card at the same time as the MAC address. Since the MAC addresses are globally unique and should not change over the lifetime of the networking card, a fixed identity key is not a problem. On the contrary, a hardware based protection of the key creates an added layer of security. Organizations with the right to distribute MAC addresses have their own Organizationally Unique Identifier (OUI) which is encoded in the first three octets of all MAC addresses distributed by this organization. Using this OUI, the public parameters needed for the MAC address can be found. This entails a very small and lightweight public parameter lookup mechanism matching OUIs to public parameters. This is the only step where any form of cooperation is needed on the organizational level, since all OUIs must be publicly available. However, since the number of OUIs is small and does not change frequently, it is easy to solve this part of the distribution. The huge benefit of this structure is that the identity key distribution can now be automated in-band in a secure fashion without relying on extensive existing security mechanisms. Using this approach, it is possible for the requesting entity to add a proof of legitimate MAC address possession using the identity key of the MAC address when requesting its IP address. This not only prevents the problem of MAC spoofing, but also allows the DCHP server to send the identity key for the IP address to the requesting entity protected with the MAC based identity encryption. Since this mechanism is only used for requesting the identity key, which is done in an Intranet, the proposed solution does not open a backdoor to the Network Interface Card producers to decrypt the Internet traffic.

6.2.3 Key Expiration

Another practical issue of network layer encryption is the fact that especially in IPv4 networks, IP addresses are reused. In a PKI or CA based IPsec solution, this creates several problems, since the central PKI must be updated or the CA must be contacted to resign public keys as the users swap IP addresses. Certificate Revocation Lists can be used to accomplish this, but the response time until a change is propagated is quite long and creates a fair amount of effort. In particular, public key caching mechanisms can lead to problems. In Figure 6.1 the dynamic IP problem is illustrated. If an entity $\mathsf{E}_{\mathsf{ID}_1}$ (the filled circle) is assigned an IP address ip_1 together with the corresponding identity key $\mathsf{d}_{\mathsf{ip}_1}$, $\mathsf{E}_{\mathsf{ID}_1}$ can not be forced to *forget* the identity key after the ip address is released. Since ip_1 will probably be reassigned to another entity $\mathsf{E}_{\mathsf{ID}_2}$ (the non-filled circle) after a reasonable amount of time, $\mathsf{E}_{\mathsf{ID}_1}$ can impersonate $\mathsf{E}_{\mathsf{ID}_2}$ during the key agreement.

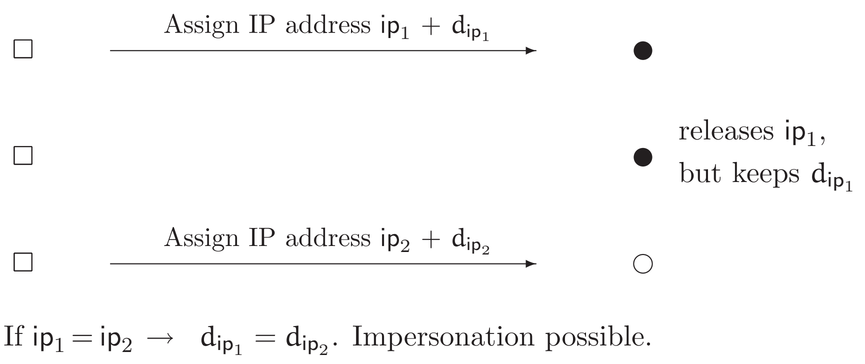

Assign IP address $\mathsf{ip}_1 + \mathsf{d}_{\mathsf{ip}_1}$
releases ip_1, but keeps $\mathsf{d}_{\mathsf{ip}_1}$
Assign IP address $\mathsf{ip}_2 + \mathsf{d}_{\mathsf{ip}_2}$
If $\mathsf{ip}_1 = \mathsf{ip}_2 \rightarrow \mathsf{d}_{\mathsf{ip}_1} = \mathsf{d}_{\mathsf{ip}_2}$. Impersonation possible.

Figure 6.1: Problem: Dynamic IP addresses

In the proposed identity-based solution, natural key expiration techniques can be used to cope with dynamic IP addresses. Boneh et al. [19] showed how keys can be given a lifetime, which allows natural expiration of the identity key. This is done by the concatenation of the ID, in this case the IP address, with a date.

E.g., the following identity key for Alice is only valid on the 20th of July in the year 1978:

$$\mathsf{ID} = \text{Alice} \rightarrow \mathsf{H}(\text{Alice}|20/07/1978)^{\mathsf{R}} \equiv \mathsf{d}_{\text{Alice}|20/07/1978} \pmod{\mathsf{N}}$$

The same technique can be used in the proposed solution. In the scenario where ISPs have a pool of IP addresses which are allocated to customers on demand and reused at will, this technique can be used such that no two customers ever receive the same identity key. Since IP address reuse is time-delayed in any case[1], this time frame can be used as the key lifetime to ensure that each successive owner lies in a new lifetime slot. With the techniques introduced in this chapter, a frequent automatic in-band key distribution can be safely executed and thus key renewal is far less of a problem. Additionally, key expiration also reduces the risk of identity key theft, since the attack window is restricted to a small time interval.

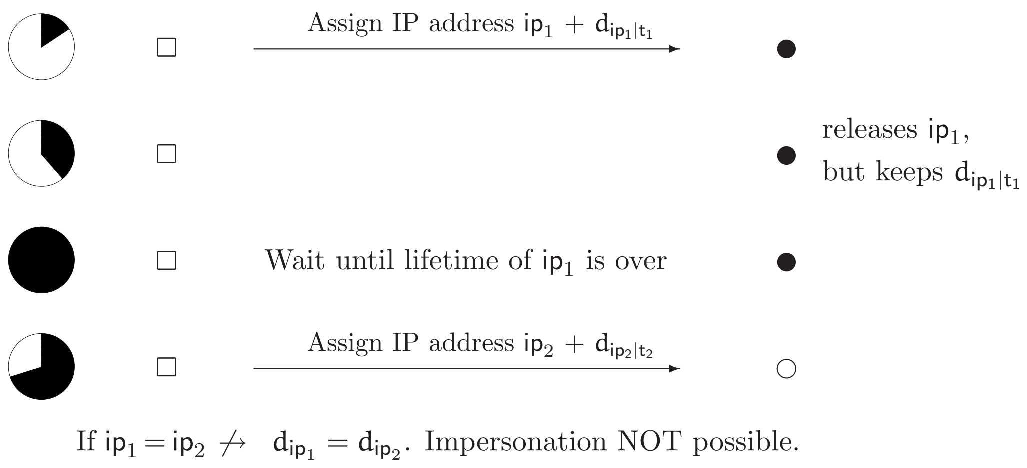

Assign IP address $\mathsf{ip}_1 + \mathsf{d}_{\mathsf{ip}_1|\mathsf{t}_1}$
releases ip_1,
but keeps $\mathsf{d}_{\mathsf{ip}_1|\mathsf{t}_1}$
Wait until lifetime of ip_1 is over
Assign IP address $\mathsf{ip}_2 + \mathsf{d}_{\mathsf{ip}_2|\mathsf{t}_2}$
If $\mathsf{ip}_1 = \mathsf{ip}_2 \not\rightarrow \mathsf{d}_{\mathsf{ip}_1} = \mathsf{d}_{\mathsf{ip}_2}$. Impersonation NOT possible.

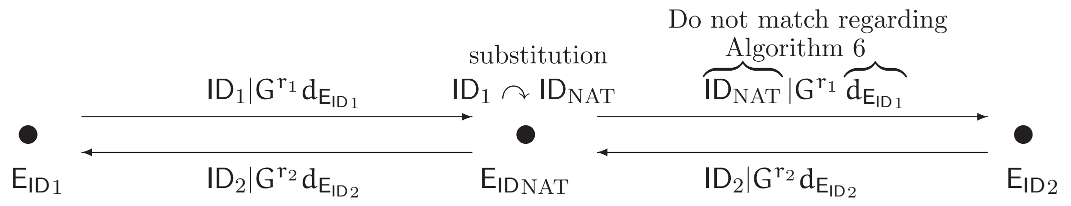
Figure 6.2: Solution: Dynamic IP addresses

The points in time, when an IP address can be reassigned to a new entity, which make up the concatenated time-stamp to the identity-key, must be known to the communication partner, since he has to consider this date within his computations. However, since these points in time are not security-relevant, they can be made public in some way.

6.2.4 NAT Traversal

The final practical issue is the NAT problem. While this mainly is a problem in IPv4 networks, there are also scenarios in IPv6 networks in which NAT is an issue. The main problem when dealing with network layer encryption when NAT is involved is that the NAT server substitutes its public IP address for the private IP address of the entity being NATed. As such, the original identity key for the private IP address is no longer valid, since it does not match the public IP address of the NAT router, hence any key agreement would fail, as illustrated in Figure 6.3.

[1]Before an IP address is allocated to a new user, a certain amount of time must pass to prevent attackers from impersonating the previous entity



Do not match regarding Algorithm 6
substitution
$\mathsf{ID}_1|\mathsf{G}^{\mathsf{r}_1}\mathsf{d}_{\mathsf{E}_{\mathsf{ID}1}}$
$\mathsf{ID}_1 \curvearrowright \mathsf{ID}_{\mathrm{NAT}}$
$\overbrace{\mathsf{ID}_{\mathrm{NAT}}}|\mathsf{G}^{\mathsf{r}_1}\overbrace{\mathsf{d}_{\mathsf{E}_{\mathsf{ID}1}}}$
$\mathsf{E}_{\mathsf{ID}1}$
$\mathsf{ID}_2|\mathsf{G}^{\mathsf{r}_2}\mathsf{d}_{\mathsf{E}_{\mathsf{ID}2}}$
$\mathsf{E}_{\mathsf{ID}\mathrm{NAT}}$
$\mathsf{ID}_2|\mathsf{G}^{\mathsf{r}_2}\mathsf{d}_{\mathsf{E}_{\mathsf{ID}2}}$
$\mathsf{E}_{\mathsf{ID}2}$

Figure 6.3: NAT - Problem

This problem is also faced by IPsec which has problems with NATed resources. When working in a NAT environment, a certain level of trust must exist between the NAT router and the NATed device. The NAT router substitutes its public IP address for the private IP address of the NATed device. The NATed device must trust the NAT router to substitute the right address, and the NAT router must be willing to forward the packets on behalf of the NATed device. However, when using encryption, the NATed device does not trust the NAT router with the plain text version of its communication. Communication between the NATed device and the outside world should still be private. Considering that the NAT router shares its public IP address with the NATed devices, the proposed solution also lets the NAT router share the identity key of its public IP address with the NATed devices (it will later be shown that this does not compromise the security of either the NAT router or the NATed devices). The identity key of its Intranet IP address is, however, kept private. Also, a private identity key is given to each NATed device, corresponding to its Intranet IP address. When a NATed device A in the Intranet establishes a connection to an external device B, it creates a SIK packet using its private value $\mathsf{G}^{\mathsf{r}_\mathsf{A}}$ in combination with the identity key of the NAT router's public IP address. This is, in essence, an extension of the normal NAT procedure to include an authenticated key exchange, and the trust relationship between the NAT router and the NATed device is not changed. The sharing of an identity key belonging to an IP address is not usual and should be avoided under normal circumstances, since anyone in possession of the identity key can pose as the legitimate owner of the corresponding IP address and thus can spoof the address or act as a man-in-the-middle attacker. However, in the NAT scenario this is exactly the desired outcome, since the NATed devices pretend to be the NAT router to the outside world, since as far as the outside world is concerned, the packets originate from the NAT router. It is important to note that although the identity key of the NAT routers' public IP address is used by the NATed device, the NAT router is not able to subvert the communication. To successfully attack the communication as a man-in-the-middle, the NAT router would also need to be in the possession of the private identity key of B, which is not the case. It is also not critical if more than one device is behind the same NAT router, since communication between the NATed devices and the NAT router is protected by the private identity key of the NAT router's Intranet IP address and the identity key of the NATed device, which is different for each device. Thus, the NATed devices are not able to subvert the communication of other devices in the Intranet nor are they able to spoof the internal identity of the NAT router or other NATed devices. Should the Intranet devices be connected to the NAT router with a pre-configured switch, the Intranet identity keys are not necessary, since the private value G^{r_A} of the key agreement is sufficient to protect the key exchange if there is a direct connection to the NAT router.

Figure 6.4 shows the solution for the NAT problem. The internal user A sends a SIK using its own private value G^{r_A} in combination with the private key of the NAT router's IP address. When the NAT router substitutes the IP address with its own, it creates a valid packet, since the value $d_{E_{ID_{NAT}}}$ now belongs to the correct source address of the packet.

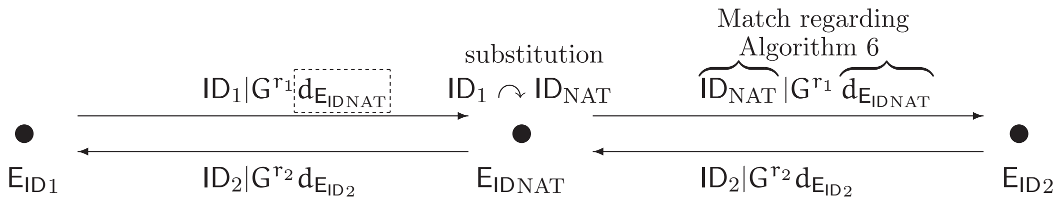

Match regarding Algorithm 6
substitution
$ID_1|G^{r_1}\boxed{d_{E_{ID_{NAT}}}}$
$ID_1 \curvearrowright ID_{NAT}$
$\overbrace{ID_{NAT}}|G^{r_1}\overbrace{d_{E_{ID_{NAT}}}}$
$ID_2|G^{r_2}d_{E_{ID_2}}$
$ID_2|G^{r_2}d_{E_{ID_2}}$
E_{ID_1}
$E_{ID_{NAT}}$
E_{ID_2}

Figure 6.4: NAT - Solution

6.2.5 SSF to Prevent IP Spoofing

Many approaches have been suggested by which IP spoofing and the resulting attacks can be prevented or at least reduced. Since it is not possible to present all of them, selected papers will be presented that cover a wide range of approaches. For a more detailed survey of techniques for IP spoofing and attacks based on IP spoofing and their prevention, the reader is referred to the paper of Peng et al. [96]. One of the most well known techniques for the prevention of IP spoofing is ingress/egress filtering as suggested by Ferguson and Senie [46]. Ingress filtering aims to prevent IP spoofing by only allowing traffic to enter or leave the network if its source addresses are within the expected IP address range. An essential requirement for this approach is to know which IP range is served by which router. For complex networks, this knowledge is hard to obtain and can change over time. One way to gain this information is reverse path filtering [8]. Since a router generally knows which IP range is reachable via its interfaces, it can check whether the return path of a packet would lie in this space. If this is the case, it forwards the packet; if not, the packet is dropped. This approach has two major drawbacks: it does not work with asymmetric routing paths, and it does not offer a personal incentive for deployment.

An approach aimed at detecting the source of spoofed IP packets during a DoS attack through IP traceback has been proposed by Savage et al. [103]. The solution probabilistically marks packets with partial path information as they arrive at routers. This approach requires that an attack comprises a large number of packets to increase the probability that an attack packet is marked. While each marked packet represents only a sample of the path it has traversed, by combining a number of such packets, the target of the attack can reconstruct the entire path. This enables the target to locate the approximate source of attack traffic without requiring the assistance of outside network operators. This can also be done post mortem as long as the packets were logged. The routing infrastructure of the Internet must be adapted to support this approach, since the routers must participate in the marking of packets. The authors state that they achieved a 99% backwards compatibility and create only a minimal overhead. However, the approach does not work if an attacker distributes the attack over a wide area or restricts the origin of the attack to parts of the network where the routers for the next hops do not mark packets. Furthermore, if an attacker can compromise a router, false traces can be inserted.

A related approach to prevent capability attacks has been presented by Parno et al. [95]. Their system is not directly aimed at preventing IP spoofing, but through the introduction of computational puzzles needed to create new connections, it limits the number of connections a single client can create. What is particularly interesting is that the puzzle is mathematically tied to the IP address of the sender, which prevents the copying of puzzles to different hosts. The initial puzzle piece is distributed via modified DNS entries. Based on the initial puzzle, the connection initiator can solve a puzzle of a chosen difficulty level. If resources become scarce, servers can prioritize clients who solve hard puzzles. This makes it difficult for a single host to deny service to others by creating thousands of connections, since the computational demand for many connections only allows easy puzzles to be solved, and thus the priority of the connections goes down. The routing infrastructure is used to take the burden off the end systems, and the existing communication infrastructure (i.e. DNS) is used to share public parameters.

Unlike the above solutions, cryptographic methods can be used to offer IP address correctness guarantees. For instance, if IPsec [69], which can be used for both IPv4 and IPv6, were to be deployed as the standard protocol for Internet traffic, IP spoofing would no longer be possible, since during the IPsec handshake, each host must use its private key to initiate the communication. Thus, the attacker would need access to the private keys for addresses that are to be spoofed. This would solve the problem of IP spoofing, but since IPsec is a heavyweight protocol designed for secure communication, it is not likely that it will be used for all traditional IP traffic anytime soon. There are several reasons. First, IPsec requires a trust infrastructure (either a PKI, a CA or pre-shared secrets) to enable communication between parties. Pre-shared secrets is a common form of setting up small IPsec environments, but does not scale up to the Internet. This leaves a CA and/or PKI approach in which private/public key pairs are tied to the IP addresses of the host computer. Setting up and maintaining this infrastructure presents a significant challenge if it is to be rolled out universally [44]. When setting up an IPsec tunnel, a session key is exchanged using asymmetric cryptography. In the case a PKI is utilized, the public key of the hosts must be accessed from the key server to decrypt the session key. In the case of a CA setup, the CA public key must be distributed to all participants. The public keys of the hosts are then signed with the CA's private key and can then be sent to the communication partner in-band, since the validity of the public key can then be verified via the CA's signature. Once the session key has been exchanged, a cryptographically secured channel can be established. The IPsec handshake requires nine messages to be sent back and forth to establish a secure channel. Currently, IPsec is used for encryption purposes, creating a high overhead for the ensuing communication as well as the handshake overhead. However, even if IPsec were to be modified to transmit in the clear when encryption is not required, the nine initiation handshake messages are too expensive to be used for the prevention of IP spoofing in scenarios where encryption is not required.

In contrast to the certificate based approaches, Liu et al.[79, 78] have presented a lightweight approach to authenticate the source of IP packets on the Autonomous System (AS) level to hinder DoS attacks and increase accountability. The proposed method uses HMAC [9] functions to create packet passports for the expected routing path of a packet. Each AS agrees on a private key with all other AS using a PKI. Once all AS involved have a separate private shared secret for all other AS, packets can be sent. To this end, the border router of the AS creates a list of HMAC values with the corresponding keys for each AS to be traversed by the packet. On route to the destination, each intermediate AS can verify the validity of the passport for its domain and thus authenticate the origin AS. This allow intermediate routers to discard packets with invalid passports, which prevents inter-AS IP spoofing and gives valid packets preferential treatment to unsigned packets in DoS situations. However, routing path changes can lead to packets being wrongly dropped due to invalid passports for the new route, and IP spoofing within the AS is not prevented, since packets receive their passports at the border gateway, and no checks within the AS are executed. For this reason, accountability is limited to the AS level.

A further approach to avoid a certificate infrastructure has been presented by Andersen et al. [5]. Here, self-certifying addresses are introduced to replace IP addresses. The Accountable Internet Protocol (AIPs) suggests to use two-level addresses containing an autonomous domain address (AD) and a unique endpoint address (EIDs). The addresses are created using a hash of the public key of the entity. To this end, the public keys must be created before the address is created and registered. New name lookup, interdomain routing and packet forwarding methods are also presented. Through the introduction of the new routing and naming infrastructure, the addresses are self-certifying. The binding of the public key to the endpoint is achieved via the hash function, and the new infrastructure allows connections to be created between the public key based endpoint.

Two similar approaches have been suggested in the context of secure IPv6 networking protocols. The Host Identity Protocol (HIP) [88] removes the need for binding IP addresses to public keys using certificates by creating a completely new form of addresses, namely HIP addresses that are constructed by hashing the public key of an entity. This creates two requirements that a HIP system must meet: (a) the public keys must be created before the address allocation can be performed and (b) a new protocol layer must be implemented between the transport and network layer that maps HIP identifiers to the routable IPv6 addresses and provides authentication. To address the latter issue, Cryptographically Generated Addresses (CGA) [7] were proposed to encode a public key into the 64-bit identifier of the IPv6 address, thus avoiding the need to change the protocol stack. However, CGA still requires the public key to be created before the IPv6 address and restricts the choice of addresses that can be used. Obviously, getting ISPs to issue particular IPv6 addresses based on user keys is a difficult task, and it remains to be seen if the effort is less than running a CA.

6.2.6 Denial-of-Service Attacks

One of the most relevant attacks based on IP spoofing is the Denial-of-Service attack (DoS) [87]. A DoS attack is an attempt to make a computer resource unavailable to its intended users. Many DoS attacks flood the target with requests to achieve this goal. To mask their attack and to make it more effective, the attackers often use IP spoofing.

6.2.6.1 Requirements

A viable approach to prevent or hinder IP spoofing and thus the described DoS attacks must fulfill the following requirements:

- The approach must be able to deal with routing path changes. Requiring a router to drop packets that do not stem from its field of responsibility reduces the scalability and robustness of the Internet and thus is not acceptable.
- The approach must be able to deal with asymmetric routing paths. Asymmetric routing paths are an important and common occurrence in large networks. Restricting their use reduces the performance of the Internet.
- The approach should not require detailed knowledge of the network topology or require large state information to be maintained.
- The approach should not require changes to the routing protocol currently in place.
- The approach should not require modification of the core network of the Internet. Any complexity added here reduces the maintainability and robustness of the Internet.
- The approach should offer personal incentives to early adopters. Many IP spoofing prevention schemes do not protect the deployer, but they protect the rest of the Internet. For instance, if an Internet Service Provider (ISP) deploys ingress filtering, the ISP is protecting customers of other ISPs from spoofed packets, but their own users are still vulnerable to spoofed packets from other ISPs without ingress filtering. The solution should offer benefits to the deployers to motivate the adoption of the technology.
- The approach should offer some benefit for partial deployment. Any mechanism that requires all parties to adopt the technology and only starts working when it is deployed across the entire Internet, will have difficulties in getting adopted.

- Finally, the complexity of the approach must not be tied to the complexity of the underlying network. The solution needs to be as simple as possible to reduce the risks of errors and misconfiguration.

6.2.6.2 Clock Synchronization

The presented signature algorithms use a timestamp that is created by the proving entity and is checked by the verifier so that it is not outdated. To enable the verifier to make a reasonable decision, the clocks in the system need to be loosely synchronized. This can be achieved with the network time protocol (NTP). Since UTC time is used, time zones do not present a problem. If transient identity keys are used (i.e. IP addresses are reused, so different users get the identity key for the same IP address but at different times), the identity key is extended by a validity period. This validity period can be used by the verifier to detect whether the sender uses faked timestamps that do not fall within the legitimate usage period. This can be achieved by using the techniques proposed by Boneh and Franklin [19] (see also Section 6.2.3). Their basic idea is to concatenate timestamps with the identities to vary the private key each time.

6.2.6.3 Puzzle Integration

An adversary interested in flooding the signature verification unit with malicious traffic would probably not follow the steps to create a valid signature because of the incurred computational costs. He would just generate random integers and send them out pretending to send a well crafted signature. The generation of such random integers would be very fast and thus an adversary could flood a verifier using these packets. To circumvent this problem, the technique proposed by Parno et al. [95] is used in SSF. The verifier could require that either the hash value of the first or the last integer of the signature must fulfill a desired property, such as the divisibility by a certain integer D. The divisibility property can be satisfied by the creator of the signature, since a random integer is involved in the generation process. The larger D is, the more time a sending unit has to spend to generate a valid signature. The generation time increases by a factor of D/2 on the average, since an adversary has to generate D/2 packets on the average, such that, for example, the last integer of the pretended signature is a multiple of D. Thus, the generation process can be easily slowed down, whereas the verification process increases only marginally for the new task to check the divisibility prior to the verification of the signature. Note that puzzles do not prevent IP spoofing, but they do hinder an attacker from using the SSF verification mechanism for a DoS attack.

6.2.6.4 Transport Layer Integration

One of the main benefits of SSF is the provable legitimate possession of an IP address without the large overhead of an n-way handshake of a full encryption protocol like IPsec. In SSF, a single packet can be used to prove the legitimate possession of an IP address. The benefit of a single packet solution is that it does not open a further denial of service attack vector (like the *syn flooding* vulnerability). The way how SSF handles to the problem of Network Address Translation (NAT) is presented in [108].

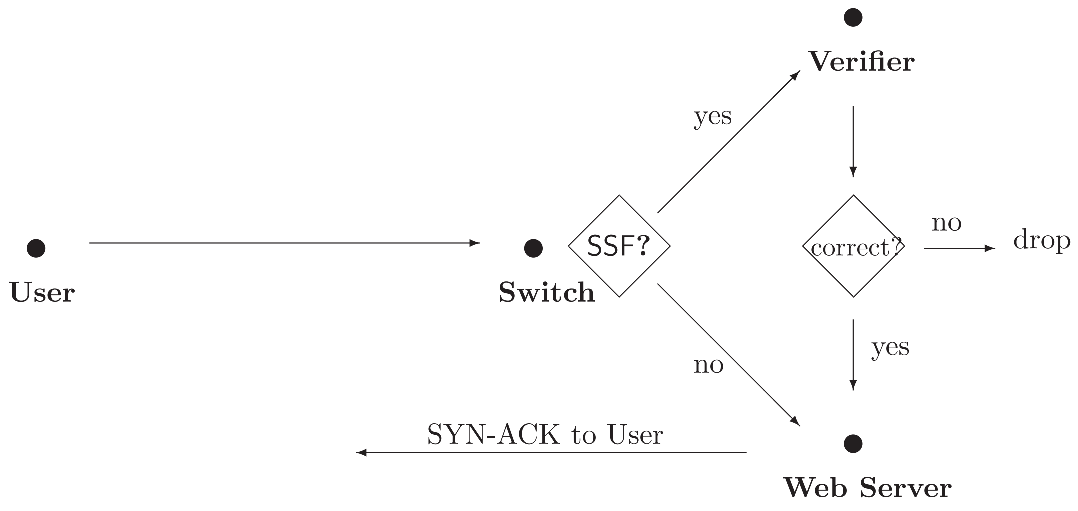


Figure 6.5: SSF setup for a TCP based application.

Since the proposed scheme requires only a single packet to prove the legitimate possession of an IP address, its integration into the UDP protocol is easily possible. The data field that carries the actual payload is used to carry the signature. UDP adopts the characteristics of the underlying network layer. In the SSF case, it is the IP layer that makes use of packets with $2^{16} - 1$ bytes at maximum. Subtracting the IP and the UDP header leaves 65507 bytes for the actual payload that can be fragmented by the IP protocol into pieces of a few kilobytes. Assuming that the modulus N has 2048 bits, it follows that the integers $\mathsf{G}^{\mathsf{T}\alpha}\mathsf{d}_{\mathsf{ID}}$ and $\mathsf{G}^{\mathsf{R}\alpha}$ are smaller and yield at most ≈ 512 bytes for the first two entries. The third entry, the timestamp, commonly is a 32-bit value, i.e. 4 bytes in size. Adding both values together results in an upper bound of about 516 bytes in the data field. Note that the proof is only inserted into the first UDP packet

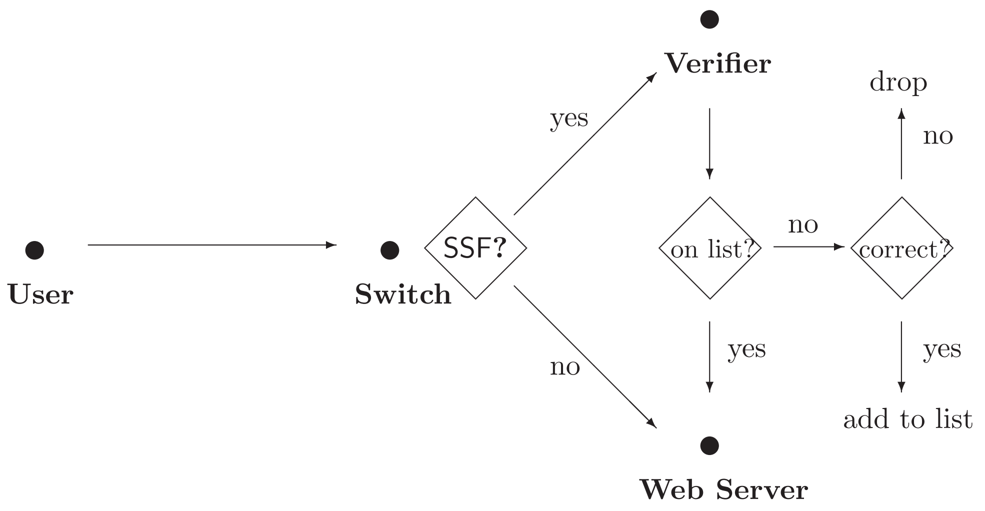


Figure 6.6: SSF setup for a UDP based application.

but not into each UDP packet, since this would create an unacceptably high overhead for many UDP applications. The first UDP packet is used to open a window for the source IP in the firewall of the destination. This simple mechanism can be used to make UDP DoS attacks based on IP spoofing more difficult, since only addresses that recently contacted the host with a valid proof can be spoofed, significantly lowering the number of spoofable addresses.

Unlike the UDP protocol, the TCP protocol is a stateful protocol that is more suitable for SSF, since only the connection initiation needs to be complemented by a proof of IP possession, because all further packets are part of the TCP stream. In this case, no special rules or lists need to be implemented by the firewall to open timed windows for further packets, since the TCP session can be used to identify legitimate packets. The data field in the TCP header is unused in the standard SYN packet, which allows to add a SSF signature there. The TCP/SYN packet is the first packet of the 3-way handshake and normally does not contain any data. The TCP RFC, however, allows TCP/SYN packets to contain data. This feature is used in SSF to piggyback a proof of IP possession into the handshake. A standard TCP/IP implementation simply ignores the data field of the SYN packet, making this approach backwards compatible.

This allows a simple integration of the proposed approach into legacy systems. If no IP possession verifier is present, the kernel will ignore the proof in the data field of the packet. If a verifier exists (e.g. integrated into a firewall), it reads the data field and validates it and forwards the SYN packet only if the signature is valid. Thus, in both cases neither the routing protocol nor the application server need to be altered. If the signature has been verified successfully, the IP address is either added to an *allow list* or a SYN/ACK is send to establish the communication. Both actions are only possible if the sender is the legitimate owner of the IP address. Session hijacking is a problem that can occur after this point; it can be handled by the usual cryptographic countermeasures.

6.2.6.5 Early Verification

In Figure 6.7, the concept of early verification is illustrated. AS 4 hosts the target server, AS 1 and AS 2 have already adopted SSF, AS 3 has not. Each verification enabled router simply drops connection requests that do not contain a valid proof, but lets proven connections requests through. AS 4 which has not adopted SSF must be blocked as usual and does not get serviced while the attack lasts. This entails that legitimate users who have not adopted SSF get their connection dropped, which of course is not desirable, but this is preferable to no users being able to use the resource at all. Furthermore, this creates an incentive to adopt SSF, since the users who sign their connection requests receive a higher grade of service and can continue using the resource during a DoS attack.

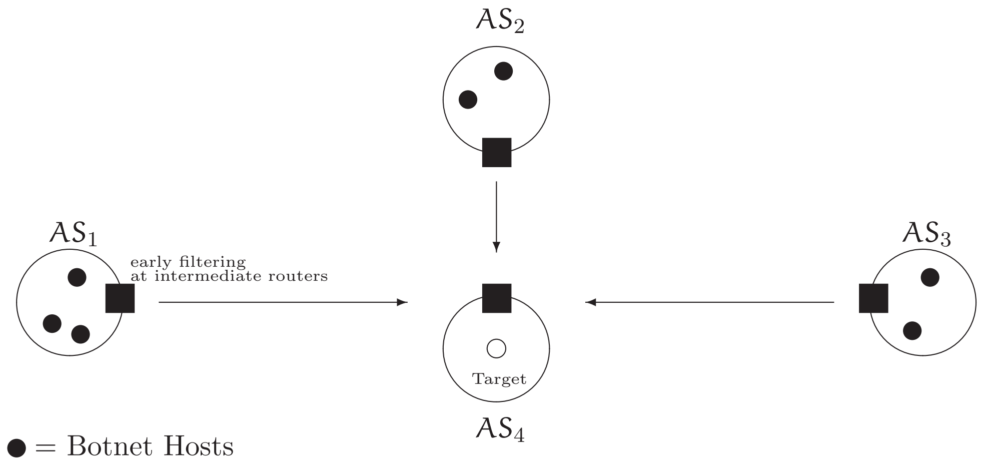


Figure 6.7: Early Router Filtering

6.2.6.6 Evaluation of Network Overhead

Apart from the cryptographic performance that is equivalent to a RSA-based approach, the network overhead is a relevant factor to be evaluated. A comparison of SSF with a standard PKI approach using X.509 certificates signed by a Certificate Authority as used in IPsec is presented in the following. To bind a public key to an IP address via a certificate, the IP address and the signature of the CA must be part of the certificate. The X.509 certificates also contain several additional parameters, like validity timestamps, the name of the issuer, the subject's name (the IP address) etc. The lengths of these additional parameters vary. Based on an analysis of the X.509 certificates used in our department for both the web servers and other activities, the average of this additional data is estimated to be $\approx$ 5000-bits. The CA key, the public key and the PKG key have a length of 2048-bits each.

Certificate Approach. To prevent man-in-the-middle attacks, the following information must be transmitted: (1) The certificate including the user's public key, the signature of the CA and the additional parameters of the X.509 certificate. (2) A signed piece of data that can be verified by a third party using the user's private key to proof legitimate possession of the public key. Thus, there are 2048-bits (user's public key) + 2048-bits (signature of CA) + 5000-bits (additional parameters) + 2048-bits (proof signature) = 11144-bits in total. The above bit length can be reduced to 6144-bits by omitting the additional X.509 parameters that could partially be hard-coded into the verification system.

SSF. In SSF, the binding between key and IP address is done implicitly by the mathematics creating the proof of possession, and thus no certificate is needed. The total bit length is: 2048-bits ($\mathsf{G}^{\mathsf{T}\alpha}\mathsf{d}_{\mathsf{ID}}$), 32-bits ($\mathsf{T}$), 2048-bits ($\mathsf{G}^{\mathsf{R}\alpha}$) = 4128-bits. This results in a reduction of factor $\approx$ 2.7 compared to the standard CA based approach, and a factor of $\approx$ 1.5 compared to a CA based approach omitting the additional X.509 parameters mentioned above.

6.3 SSF to Secure Phone Calls

The proliferation of mobile telephones is extensive, with billions of handsets in active use in almost all countries. However, unlike the area of network security, mobile phone call security is severely lacking. In mobile phone networks, eavesdropping on a call is easy, even for non-governmental forces. Since the encryption schemes in GSM (2G) and UMTS (3G) only encrypt calls between the mobile phone and the base station, an attacker positioned anywhere in the network between the two base stations can usually intercept calls without great difficulty. Furthermore, since GSM base stations are not authenticated, an attacker can pose as a base station and intercept phone calls in the vicinity. Due to backwards compatibility and UMTS coverage issues, most UMTS devices allow network fallback to GSM, opening up UMTS devices to the same man-in-the-middle attacks that afflict GSM networks. While it is possible to implement end-to-end encryption of mobile phone calls based on a Public Key Infrastructure (PKI), the complexity of setting up and using a PKI is prohibitive, especially since many users of mobile phones are not well versed in cryptographic procedures and are quickly overwhelmed when confronted with public and private keys, certificates, signatures and revocation lists.

Identity-based cryptography (IBC) promises to offer an approach to end-to-end encryption for mobile telephone calls in which the telephone numbers of the call participants are used as the public keys to secure the communication channel, thus making the cryptographic security procedure as easy as making a telephone call. The use of telephone numbers as public keys has two major benefits. Firstly, since the caller knows the number to be called, the caller also automatically knows the public key and does not need a separate public key look-up or certification infrastructure. Secondly, telephone numbers are easy to understand and users are confident in using them, such that there is no need to educate users to understand the link between a telephone number, a public key and/or its certificate, thus significantly lowering the complexity threshold of phone call encryption.

6.3.1 GSM

In GSM networks, communication between a mobile system (MS) (i.e. a mobile phone) and a base transceiver station (BTS) is encrypted using the A5 [98] cryptographic protocol. Due to design flaws, A5 is vulnerable to cryptanalysis such that hackers can eavesdrop on the communication. Updates to the A5 protocol have been proposed to hinder further attacks, and the UMTS standard has replaced A5 by a more secure (and open) protocol, making cryptographic attacks less of a concern. A simpler attack is to subvert the communication setup before encryption. To allow a MS to authenticate itself to the network provider, it gets a subscriber authentication key (SAK). The SAK is stored both on the SIM card of the MS and in the Home Location Register (HLR) of the provider. The BTS are connected to a Base Station Controller (BSC) that in turn is connected to a Mobile Switching Center (MSC) and a Visitor Location Register (VLR). These in turn are connected to the HLR and the Authentication Center (AuC) that give access to the SAK of the MS. During the authentication process, a 128-bit random number is generated which using the A3 [34] is combined with the SAK to create a 32-bit authentication key called SRES. The SRES key is then sent to the BTS. The SRES key is then compared to the SRES* key that is computed by the AuC of the provider also using the A3 algorithm and the HLR SAK. If the two values match, the MS is authenticated and may join the network. The BTS does not authenticate itself to the MS. This opens up the possibility of a Man-in-the-Middle (MITMA) attack.

Using an IMSI catcher, an attacker can pose as a BTS and intercept calls in the vicinity by broadcasting a strong base station signal. MS are programmed to connect to the strongest BTS signal, thus if the IMSI catcher has the strongest signal they serve their current BTS connection and will connect to the IMSI catcher no questions asked. Since the BTS is also responsible for selecting the security mechanism, the IMSI catcher can then force the MS to turn off or select an insecure encryption algorithm and thus allow the MITMA to operate. The downside to this attack is that the IMSI catcher cannot function as a real BTS since it is not connected to the main phone network and must forward calls using its own MS and SIM.

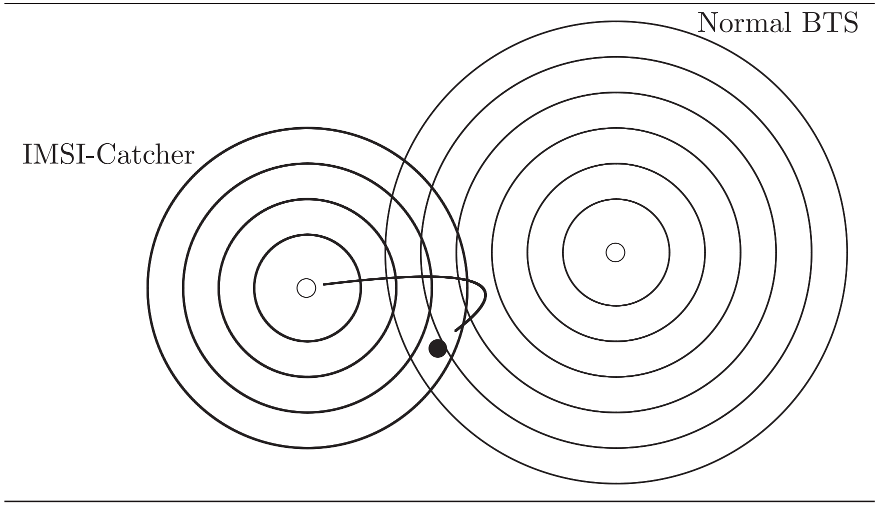


Figure 6.8: Because of its stronger signal (properly due to its local closeness), the IMSI-Catcher forces the cellphone to register at him rather than the original BTS. After negotiating a non-encrypted communication, the IMSI-Catcher forwards and eavesdrops all packets.

However, since the SIM in the IMSI catcher cannot register itself as the target SIM (due to the authentication of the MS), the attacked MS is not registered at any BTS and is not reachable while it is connected to the IMSI catcher. Thus, only outgoing calls can be intercepted, since the network cannot reach the attacked MS. Furthermore, the IMSI catcher is not a targeted attack. It affects all MS in its vicinity all of which

are not reachable while they are connected to the IMSI catcher and whose calls would need to be forwarded if the IMSI catcher is not to become noticeable. While this attack should not be taken lightly, there are some real world problems in its execution.
A much simpler attack is enabled by cost saving measures in common practice when setting up base stations. Since connecting all BTS to a secured wired network is costly, BTS can also be connected to the main network via a directed microwave link. This microwave signal is sent without encryption and can easily be intercepted, giving an attacker clear text access to all calls going via this link without leaving a physical trace. But even a wired connection is not safe if an attacker is willing to apply a physical tap to the line. These link taps are particularly relevant since they can be used without affecting the rest of the network and thus cannot be easily detected. They also allow a large number of calls to be tapped simultaneously. For instance, a BTS located near a firm, government building or celebrity house can be tapped, thus, making all mobile calls made to and from that location available to the attacker. Since the equipment needed to execute such a tap is becoming more portable and cheaper at a rapid rate, this kind of attack will rapidly gain in relevance.
To prevent the above attacks, end-to-end protection of phone calls is required. However, the solution must be able to be deployed in a multi-organization environment and be usable by non-tech savvy users. As stated in the introduction, conventional PKI based solutions are too complex both for the network providers and for the users. A simple approach is required which can be implemented by network providers independently of each other and which does not introduce added complexity for end users.

Zimmerman's Protocol: ZRTP. It is easy to understand that telephony is a perfect scenario for IBC since the remote identity (= telephone number) has to be known. However, there is another approach which has become very popular in the last two years which is called ZRTP [128]. ZRTP is an extension of the plain Diffie-Hellman key-agreement protocol but **not** identity-based. ZRTP is restricted to telephony or other type of communications forms where the two involved party can directly *hear* each other. The users check the status of their encryption via a short authentication string (SAS) which the users read and verbally compare over the phone. The SAS will only be equal, if the key-agreement was successful. All further communications between the same parties are secured via derived keys from the initial session key. Therefore, each ZRTP endpoint maintains a long-term cache of shared secrets that it has previously negotiated with the other party.

The security concept of ZRTP is based on the fact that the participants compare a derived value from their actual encryption key over the voice channel. It is clear that

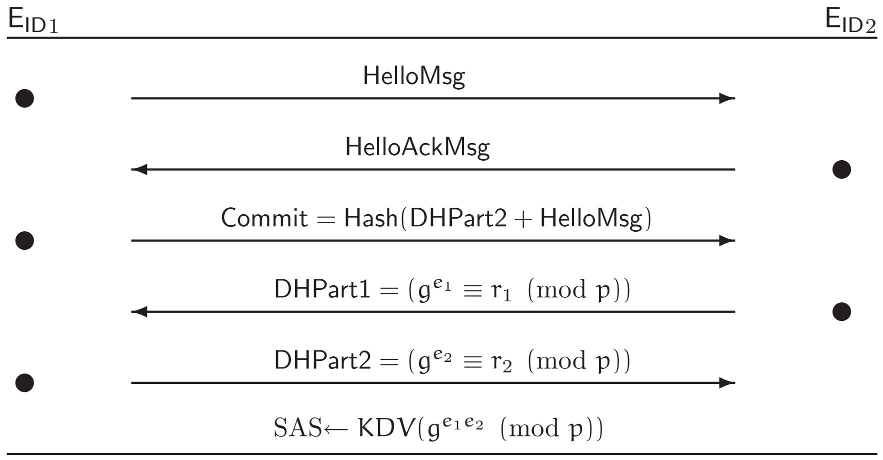

E_{ID1}
E_{ID2}
HelloMsg
HelloAckMsg
Commit = Hash(DHPart2 + HelloMsg)
DHPart1 = ($g^{e_1} \equiv r_1 \pmod{p}$)
DHPart2 = ($g^{e_2} \equiv r_2 \pmod{p}$)
SAS← KDV($g^{e_1 e_2} \pmod{p}$)

Figure 6.9: The ZRTP protocol flow (with no available preshared/previous secret keys).

this will fail if no voice channel or the like is available, which hinders ZRTP to be a generally applicable solution. On the first sight, it seems, that by the comparison of the SAS via the voice channel, the security of the encryption process is completely reduced to the mathematical problem of the DH key agreement. But this is not true. In [97] it is explained how ZRTP can be attacked without touching the mathematical process. For example, speech synthesizers are a promising option for an attacker. They can be used to insert a fake speech block, which contains a SAS, that is spoken with the voice of the intended participant. Even worse, most of the time, probably only one participant will read the SAS string whereof the other one will simply acknowledge the correctness with a short "'yes"'. The word yes can be even easier synthesized than all possible SAS strings, which makes such an attack more practical.

A Perfect Scenario

In the sequel, a scenario is shown in which the GSM world is aligned to the application of identity-based encryption.

We start with the construction of the cell phone. The public shared parameters of the main providers can already be stored in memory at this point, likewise the certificates of the main root CAs come with the installation files of a web browser.

Because the phone number is determined at a later point in time, a private key cannot be computed at this stage.

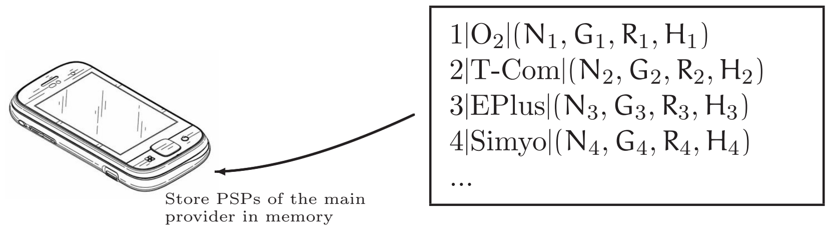

Store PSPs of the main provider in memory
$1|O_2|(N_1, G_1, R_1, H_1)$
$2|\text{T-Com}|(N_2, G_2, R_2, H_2)$
$3|\text{EPlus}|(N_3, G_3, R_3, H_3)$
$4|\text{Simyo}|(N_4, G_4, R_4, H_4)$
...

Figure 6.10: The main providers PSP are stored within the cell phone during construction.

After a customer orders a cell phone, a telephone number is chosen. Concurrently, the key generator of the associated provider generates the private key for the corresponding number and stores these information on the SimCard (see Figure 6.11). In this way, the private key is not bound to a mobile phone, but to a telephone number as it is supposed to be.

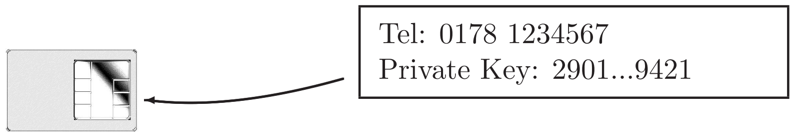


Figure 6.11: The private key for the associated telephone number is stored on the Simcard, whenever it is sold to a customer by a provider.

Whenever two participants from different providers want to communicate, the procedure is as follows (see Figure 6.12). The caller *Alice* selects the callee, here *Bob*, from the contact list in her cell phone. The prefix of his telephone number tells that he is T-Com customer. Since Alice herself uses O_2, she has to extend her private key as well as the hash value of the Bob's telephone number to make the key agreement possible.

After executing the Extend algorithm, Alice can build the session initiation key, which she sends to Bob during the connection establishment. Bob himself sees Alice calling and computes the extended keys on his part in the analog way.

6.3.2 Voice over IP

Telephony over the Internet or Voice over IP (VoIP) has earned much attention in the last years and consequently gains market shares. However, beside the financial savings,

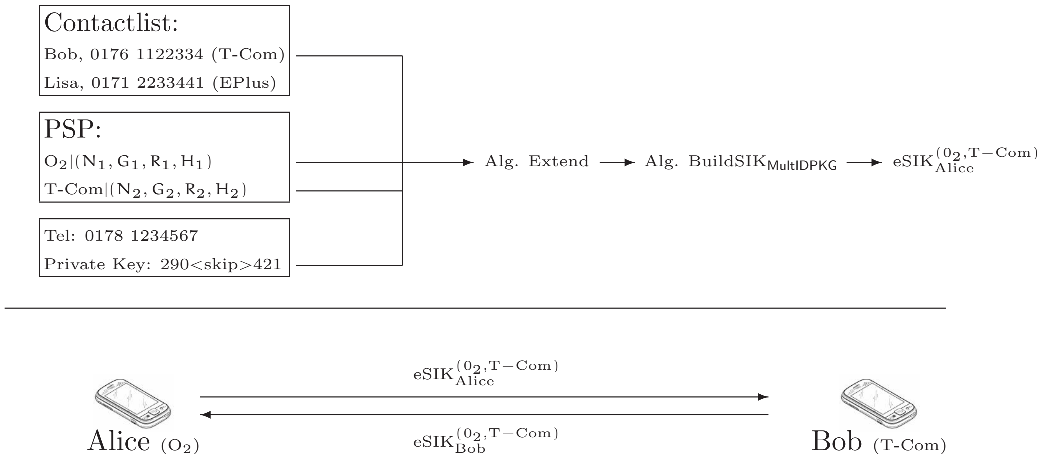

Contactlist:
Bob, 0176 1122334 (T-Com)
Lisa, 0171 2233441 (EPlus)
PSP:
$O_2|(N_1, G_1, R_1, H_1)$
T-Com$|(N_2, G_2, R_2, H_2)$
Tel: 0178 1234567
Private Key: 290<skip>421
Alg. Extend
Alg. BuildSIK$_{\mathsf{MultIDPKG}}$
$eSIK_{Alice}^{(O_2,T-Com)}$
$eSIK_{Alice}^{(O_2,T-Com)}$
$eSIK_{Bob}^{(O_2,T-Com)}$
Alice (O_2)
Bob (T-Com)

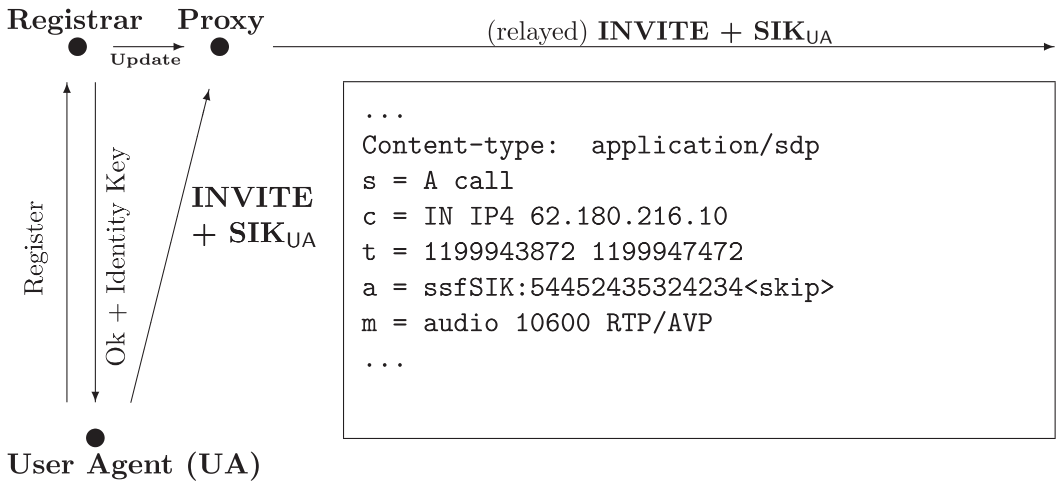
Figure 6.12: Communication.

the risk of being eavesdropped increases a lot. In the last years, VoIP threats were placed at the top of the lists of IT-security risks.

Around the *de facto* standard protocol for VoIP session establishment, the SIP protocol (SIP: session initiation protocol), many RFC proposals have been made. Two of them are widely accepted; the RFC 3711 SRTP and the RFC 3830 MIKEY. The first one is an extension of the of RTP protocol, which is one of the most popular protocol in this area. RTP expects a symmetric encryption key that must be known to all participants to encrypt the entire communication. This key distribution is mostly done by MIKEY, the latter one of the two RFCs. Therefore, MIKEY allows three modi: Pre-shared keys, Diffie-Hellman and public key infrastructures.

All of these are armed with disadvantages. The pre-shared modus is generally non-applicable for ad-hoc communication, Diffie-Hellman is non-authenticated and PKIs causes a heavy overhead. A good approach would be to extend these modi by another option that is based on IBC. Regarding implementation, this can be done using SIP Extensions.

In Figure 6.13, an example connection establishment is shown when using SSF in a VoIP environment based on SIP Extension.



```
...
Content-type:  application/sdp
s = A call
c = IN IP4 62.180.216.10
t = 1199943872 1199947472
a = ssfSIK:54452435324234<skip>
m = audio 10600 RTP/AVP
...
```

Figure 6.13: VoIP: Registering and Calling.

6.3.3 Implementations

As a reference implementation, a dll-library written in `c++` was created. It contains all algorithms of the proposed scheme. Using this library, the key agreement protocol was integrated into the following applications:

- **Jabbin**: The proposed scheme was integrated in the VoIP-Softphone Jabbin (*www.jabbin.com*), which is based on the XMPP-Server OpenFire (*www.ignite-realtime.org*) and the P2P-Network implementation LibJingle from Google. Due to the XML-based messages in LibJingle, the actual negotiation protocol could easily be extended by a new attribute, that is the session initiation key. This makes the extended version even compatible with non-SSF version, since if the SIK attribute is absent, the applications switches to the normal insecure mode.

- **Wengo**: The proposed scheme was integrated in the VoIP-Softphone Wengo (*www.openwengo.org*) based on the OpenSource SIP-Server MJSip. For this purpose, the session initiation protocol was equipped with new attributes via SIP Extensions. Due to the existing support of the plain Diffie-Hellman key agreement, a simple substitution of the DH public key with the SSF session initiation key could be made. All this was done in the SIP message type `INVITE`. The actual implementation was done by Graf [56].

- **Symbian (N95)**: The proposed scheme was integrated in the Symbian OS 9.2 FP1 using a lightweight version of our dll. The SSF algorithms itself ran well on the phone and outgoing calls could be established. Because of provider problems regarding the necessary data channel, the implementation was not completely productive at the end. However, the practicability was shown. The actual implementation was done by Agel [2].

6.4 Optimizations

Cryptographic protocols that use public/private keys are typically based on NP-hard mathematical problems. To make such a problem infeasible to compute on modern hardware, the involved integers or dimensions must be sufficient large. This leads to computations with very large integers, which are very expensive regarding computational power and sometimes memory consumption. This extra cost can extend the time to set up a secure channel significantly. A simple RSA encryption with an up-to-date key size can take several seconds of time on a PDA, whereby the CPU utilization reaches its maximum.

The most costly operation in this domain is exponentiation. Exponentiation with secure bit sizes requires several hundred or thousands of multiplications, consuming quite some time on small devices. Compared to the size of memory or the available bandwidth, the actual bit size of the involved integers is relatively small, e.g. a 1024-bit exponent does not need much space to store or much time to transfer, but used as an exponent it entails much effort.

A portable device is always or most of the time connected to a network. Thus, an interesting idea is to “outsource” expensive computations by submitting the relatively few bits to a computationally powerful backend server $\mathcal{B}$. Cryptographic operations sometimes contain sensitive information such as a private key, but sometimes they include just a multiplication with publicly known integers. In the latter case, outsourcing is without a risk, since no sensitive data can be stolen. The only damaging case that can occur is when the backend server always responds with false results, making all further computations needless. In the case when private information is involved, this information must be *blinded* such that the remote server cannot extract useful information from it. Obviously, the blinding operation itself should be less in cost than the actual cryptographic operation in question. Figure 6.14 illustrates the task when two participants use a key agreement protocol that enables outsourcing during the involved steps. In step 1, participant 1 (the initiator) builds a packet with the help of the backend server that sends the aided computation back in step 2. During step 3, participant 2 receives the packet and uses the server to handle the received packet (steps 4 and 5) and to create its own packet (steps 6 and 7). In step 8, participant 2 responds. In step 9 and 10, participant 1 uses the backend server to process the received answer.

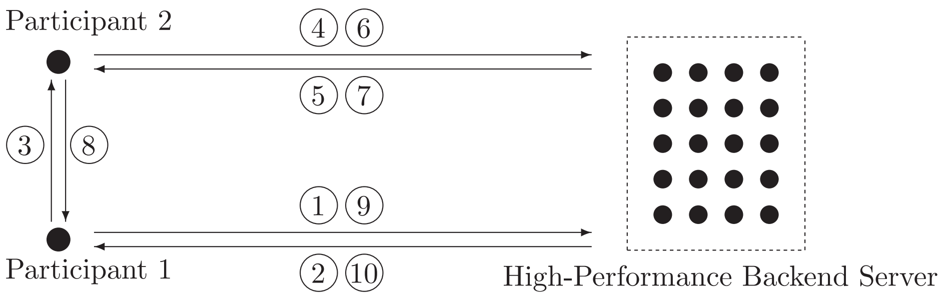


Figure 6.14: A protocol using a compute cluster for cryptographic computations.

6.4.1 Server-aided Cryptography

The aim of server-aided cryptography is to support small devices in expensive operations. This is done by delegating these operations to powerful servers that do the computations on behalf of the owner of the small device.

Here, the Algorithms 7 and 8 as well as the corresponding Algorithms 10 and 11 would be candidates to be outsourced. These algorithms perform an exponentiation that involves an exponent of non-negligible size. In the rest of this Chapter, it is focused on Algorithms 7 and 8 only.

When using server-aided computations, it has to be investigated whether the utilized server can be trusted or not. If the server cannot be trusted, the computational task may be performed wrongly by the server in order to harm the client. In such cases, the client has to verify the result in some way. If the server can be trusted, the client can assume that the computation is done as demanded and the result does not need to be verified. However, trusted or not, in no case the server is allowed to obtain a secret key that can be involved in the computation. For example, if the task is to compute G^{r} where r is a secret key, the server is not allowed to access the integer r at any time. In such cases, the secret key has to be hidden or blinded.

It is assumed that the backend server $\mathcal{B}$ can be fully trusted and the packets pass the network connection untampered. In proposed case, this is not unrealistic, since a provider who wants to support its clients with server-aided computations is interested in satisfying its customers. In this case, the Algorithm 7 can be outsourced as shown in Figure 6.15.

Setting: Both participants possess the PSP = $\{N, G, R, H\}$
Preprocessing: (This is done only once for each mobile device)
1. E_{ID} computes a random integer ρ
2. E_{ID} computes $s \equiv G^{-\rho} \pmod N$
3. E_{ID} stores (ρ, s)

E_{ID} $\mathcal{B}$

● $r \overset{rand}{\leftarrow} \{2^{k-1}, 2^k - 1\}$
● $\hat{r} = r + \rho$
● $\xrightarrow{\hat{r}}$
compute $t \equiv G^{\hat{r}} \pmod N$ ●
$\xleftarrow{t}$ ●
● compute $\mathrm{SIK}_{ID_1} \equiv s \cdot t \cdot d_{ID_1} \equiv G^{-\rho} G^{r+\rho} d_{ID_1} \equiv G^r d_{ID_1} \pmod N$

Figure 6.15: Outsourcing of Algorithm 7

E_{ID} adds a random integer ρ to the actual exponent, which hides the exponent from disclosure. In the end, E_{ID} subtracts the integer by using a precomputed integer. It is clear that $\hat{r}$ releases almost no information about the real integer r. The computational amount reduces from one exponentiation with $\mathcal{O}(\log_2 r)$ number of multiplications to one addition and two multiplications (apart from the preprocessing step that only has to done once for a device). Despite the additional cost of transferring the necessary bits, this is a large speedup compared to the original runtime. This method has already been proposed by Lim and Lee [77].

The reason why this simple but effective method works is that the base G is fixed, which makes the preprocessing step possible. On the contrary, in Algorithm 8, things are different. Here, the base $(\mathrm{SIK}_2^R \cdot H(ID_2)^{-1}) \equiv G^{R r_{ID_2}}$ is not known in advance. SIK_2 as well as $H(ID_2)$ are both values that depend on the corresponding participant, thus precomputation becomes impossible in this case. Computing the value $G^{-\rho}$ on the fly would lead to the same computational effort as computing G^r directly. It is focused on the exponentiation with $2r_1$ in Algorithm 8 only, since the inner exponentiation SIK_2^R can be neglected due to the fact that the integer R can be chosen very small.

An additional problem in Algorithm 8 is that the order of G in $\mathbb{Z}_N$ as well as the factorization of N is unknown. Consequently, the server-aided computation techniques proposed in the literature [76, 77, 41, 63, 81] cannot be used. In particular, a satisfactory solution for speeding up exponentiation by server-aided methods, where the base as well as the exponent is random and secret, has not been published yet. All existing methods make use of the property that either the order or the base is known or that the exponent need not be kept secret.

Matsumoto et al. [81] have proposed an approach to let a server compute the integer $x^d \pmod{N}$, where d and x can be secret. However, their approach utilizes the fact that the client knows the factorization of N. Lim and Lee [76] have presented an optimization of Matsumoto's algorithm, but do not remove the need for knowing the factorization of N. In a further paper [77], Lim and Lee focus on exponentiation modulo p, which makes the order of the involved group always known. In 2005, Hohenberger and Lysyanskaya [63] presented some new ideas to generic outsourcing methods, but they also focus on computations modulo p only, as Nguyen et al. do [90]. Dijk et al. [41] mention to do exponentiation modulo a composite number N, but they also require that the client knows its factorization.

Thus, none of these approaches considers to compute a discrete exponentiation modulo a composite integer N and its factorization is not known to either the client or the server. Additionally, the base as well as the exponent are random, and the exponent has to be kept secret.

A trivial approach would be to split the secret exponent into two parts $r = r_1 + r_2$ and submit the exponentiation task to two *different* backend servers. One backend server computes G^{r_1}, and the other one computes G^{r_2}. If the servers do *not* cooperate, they do not gain information about r and the result can simply be obtained by $G^{r_1} \cdot G^{r_2} \pmod{N}$. However, this cannot be assumed.

The proposed solution makes use of the Repeated-Squaring Algorithm typically used for discrete exponentiation. The algorithm is illustrated in Algorithm 24. It can be seen that in each round of the for-loop, at most one multiplication is performed and two if the i-th bit position of the exponent is equal to 1.

The general idea of the proposed solution is as follows: Assume $\mathcal{B}$ knows N and G. In each round, $\mathcal{B}$ computes two integers. For the first integer, $\mathcal{B}$ assumes that the i-th bit position in r is equal to 0, thus $\mathcal{B}$ skips Line 4 in Algorithm 24. For the second integer, $\mathcal{B}$ assumes that the i-th bit position is equal to 1, thus $\mathcal{B}$ performs both exponentiations, in Line 3 and Line 4. Afterwards, $\mathcal{B}$ submits the two integers to E_{ID}. E_{ID} chooses the correct one, since he knows the exponent and thus knows if the i-th

Algorithm 24 Exponentiation: Repeated Squaring

```
Input: $G, r = \sum_{i=0}^{n-1} c_i 2^i, N$
Output: $G^r \pmod{N}$
1. $a \leftarrow 1$
2. for $i = n - 1$ to $0$ by $-1$
3.      $a \leftarrow a * a \pmod{N}$
4.      if $c_i = 1$ then $a \leftarrow a * G \pmod{N}$
5. return $a$
```

bit is zero or not. After choosing the correct integer, he blinds it by adding a random integer M of sufficient size and sends it back to $\mathcal{B}$, which proceeds in the same way for the next bit position. The blinding operation can be reversed by $\mathsf{E_{ID}}$ by reducing modulo M.

After having explained the general idea, more details are necessary to prove that the algorithm indeed computes the correct result and that $\mathcal{B}$ is not able to obtain r with a non-negligible probability in the size of r. First of all, the complete algorithm is given in Algorithm 25.

Algorithm 25 Outsourced Version of Algorithm (SSF) Compute

```
Input: $G, r = \sum_{i=0}^{n-1} c_i 2^i, N$
Output: $G^r \pmod{N}$
1. compute $G_1 \equiv G^2 \pmod{N}$
2. compute $G_2 \equiv G_1 G \pmod{N}$
3. $M_{n-1} = N$
4. for $i = n - 2$ to $0$ by $-1$
5.a     if $c_i = 0$ then $W_{i+1} \leftarrow (G_1 \pmod{M_{i+1}}) \pmod{N}$
5.b     else $W_{i+1} \leftarrow (G_2 \pmod{M_{i+1}}) \pmod{N}$
6.      $M_i \overset{rand}{\leftarrow} \{N^3, N^3 + \Delta\}$
7.      $W_i \leftarrow W_{i+1} + M_i$
8.      send $W_i$ to $\mathcal{B}$
9.      receive $(G_1, G_2) \leftarrow (W_i^2, W_i^2 G)$
10. return $W_i \pmod{M_i}$
```

After skipping the most significant bit (since it must be equal to 1), the algorithm parses the exponent from the most significant bit to the least significant bit. For each bit (in Line 6), a random integer is chosen, which is added to the previous result in Line 7. $\mathcal{B}$ computes the square of W_i as well as the square of W_i times G. $\mathsf{E_{ID}}$ reduces the results first regarding M_{i+1} and then regarding N (Line 5a/b), which reveals the correct solutions in $\mathbb{Z}_N$. Figure 6.16 shows the communication and computation steps in a single for-loop iteration.

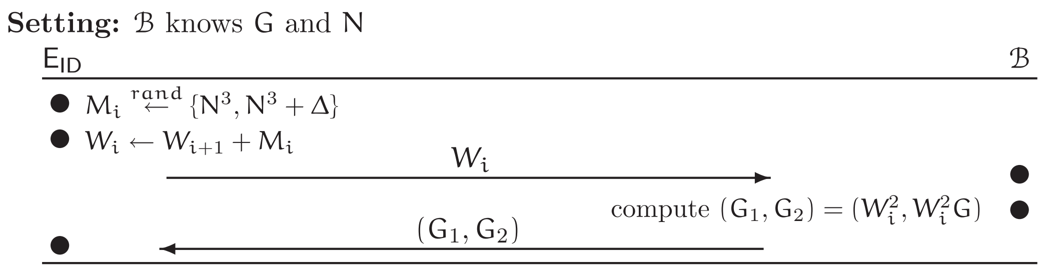

Setting: $\mathcal{B}$ knows G and N
$\mathsf{E_{ID}}$
$\mathcal{B}$
$M_i \overset{rand}{\leftarrow} \{N^3, N^3 + \Delta\}$
$W_i \leftarrow W_{i+1} + M_i$
W_i
compute $(G_1, G_2) = (W_i^2, W_i^2 G)$
(G_1, G_2)

Figure 6.16: Outsourcing of Algorithm 8

Example: Let $N = 6499$, $G = 17$ and the exponent $r = 11 = 1011_2$. Thus, the exponent is 4 bits long, and the for-loop runs from 2 to 0, as shown in Figure 6.17.

i	G_1	G_2	W_{i+1}	W_i	M_{i+1}	M_i	bit
-	289	4913	n/a	n/a	6499	n/a	$\mathbf{1}011_2$
2	W_i^2	$W_i^2 17$	289	21580589148265	6499	21580589147976	$1\mathbf{0}11_2$
1	W_i^2	$W_i^2 17$	3075	207332088832074	21580589147976	207332088828999	$10\mathbf{1}1_2$
0	W_i^2	$W_i^2 17$	5858	26566610735587	207332088828999	26566610729729	$101\mathbf{1}_2$

Figure 6.17: Example for Algorithm 25

The first row shows the initialization step. Here, G^2 and G^3 are computed with respect to the first bit, and also M_3 is set to N. All further computations of W_i^2 and $W_i^2 G$ are done by $\mathcal{B}$. The M_i values are just random integers. The final output is 5858, which indeed is $17^{11} \pmod{6499}$.

Correctness and Security. Let r be an n-bit integer. For the correctness it has to be shown that the partial result in each round is equal to the term $G^{\lfloor r/2^i \rfloor} \pmod{N}$, which is the usual partial term in the Repeated-Squaring Algorithm.

Lemma 6.4.1 *Algorithm 25 is correct.*

Proof 6.4.2 *It can be argued via induction: Since $\mathsf{E_{ID}}$ knows the correct partial result for the most significant bit, which is G itself, it can be assumed that the $(n-1)$-th partial result has been obtained by ($G^{\lfloor r/2^{n-1} \rfloor} = G$). Thus, $\mathsf{E_{ID}}$ already has $W_{i+1} \equiv G^{\lfloor r/2^i \rfloor}$ $\pmod{N}$ as the correct partial result. After submitting $W_{i+1} + M_i$ to $\mathcal{B}$, $\mathsf{E_{ID}}$ receives*

$$W_{i+1}^2 + 2W_{i+1}M_i + M_i^2 \tag{6.1}$$

$$W_{i+1}^2 G + 2W_{i+1}GM_i + GM_i^2 \tag{6.2}$$

Since $W_{i+1} < N$ and $G < N$ it holds $W_{i+1}^2 < W_{i+1}^2 G < N^3 < M_i$. Thus, Equation

(6.1) reduces to $G^{2\lfloor r/2^i \rfloor}$ *and Equation (6.2) reduces to* $G^{2\lfloor r/2^i \rfloor+1}$.

The next partial term is $G^{\lfloor r/2^{i-1} \rfloor}$*, which is either* $2\lfloor r/2^i \rfloor$ *or* $2\lfloor r/2^i \rfloor + 1$*, depending on whether the* i*-th bit is 0 or 1. Since* E_{ID} *knows this bit value, he can choose the correct value from the two possibilities, thus obtaining the correct next partial result.* ***q.e.d***

Lemma 6.4.3 *During Algorithm 25,* $\mathcal{B}$ *does not obtain the secret integer* r *except with a negligible probability.*

Proof 6.4.4 *To learn the exponent* r*,* $\mathcal{B}$ *must know whether the bit at the current position is 0 or 1. In the each round,* $\mathcal{B}$ *computes both cases; one integer for the case that the bit is 0 and one for the 1-case. Obviously, in this round* $\mathcal{B}$ *does not learn anything about the bit value. In the next round,* $\mathcal{B}$ *receives one of the previously computed integers added with an random integer* M_{i-1}*. If* $\mathcal{B}$ *could decide which integer is involved in this packet,* $\mathcal{B}$ *knows which integer* E_{ID} *has chosen, thus* $\mathcal{B}$ *knows the corresponding bit value. However, since* M_{i-1} *is chosen completely randomly and independent of previous rounds, both possibilities are still equally likely, thus* $\mathcal{B}$ *can not do better than random guessing on each bit position, which is negligible in the length of* r*.* ***q.e.d***

6.4.2 Performance Gain

To measure the performance gain, the savings of the computational cost as well as the additional overhead of the outsourcing procedure are counted. By looking at Algorithm 25, one gets the following costs: There are two multiplication at the beginning that compute G^2 and G^3, both of which require time t_{mult}. Next, there are two modular reductions in each round (Line 5a/b), one modulo M and one modulo N. The total time for these two reductions is denoted by t_{red}. In Line 6, a random integer is chosen from a given interval. Since N is a fixed public parameter, this random integer can be precomputed, thus the time required for choosing the random integer is not considered. The last computational operation in the for-loop is the addition $W_{i+1} + M_i$. Addition is a cheap operation and is often treated as getting it for free. However, the time for a single addition is denoted by t_{add}. Finally, there is another single reduction of modulo M_{i+1}. The total cost is

$$2t_{mult} + (n-1)(t_{red} + t(M)_{add}) + t(M)_{red} \tag{6.3}$$

Since multiplication is much more expensive than reduction and addition, the cost is

$$2t_{\mathsf{mult}} + (n-1)(t_{\mathsf{red}} + t(M)_{\mathsf{add}}) + t(M)_{\mathsf{red}} < \mathcal{O}(n)t_{\mathsf{mult}} \tag{6.4}$$

where the right side is the average cost for a multiplication with the Repeated-Squaring Method. Consequently, the algorithm yields a large computational saving compared to the standard case.

Next, the number of bits that need to be transferred during the algorithm are counted. Since G and N are fixed PSP, it is assumed that they are already known by $\mathcal{B}$. During one loop (see Figure 6.16), the terms $W_{i+1} + M_i$, $(W_{i+1} + M_i)^2$ and $(W_{i+1} + M_i)^2 G$ are exchanged. The bit length of M_i is $\log_2 M_i$, and since M_i is much larger than W_{i+1} (remember that $W_{i+1} < N$, since it was reduced modulo N in Line 5a/b), the bit length of $W_{i+1} + M_i$ is $\approx \log_2 M_i$. The same argument holds for the second term. The squaring doubles the bit length, thus $|(W_{i+1}+M_i)^2| \approx 2\log_2 M_i$. In the last term, the factor G has been multiplied, which adds $\log_2 N$ bits to the previous term. In total there are

$$\#\mathsf{bits} = \mathcal{O}\left(\log_2 r \left(\log_2 M + (2\log_2 M) + (2\log_2 M + \log_2 N)\right)\right) \tag{6.5}$$

bits that need to be transfered, which can be simplified to $\#\mathsf{bits} = \mathcal{O}\left(16\log_2 r \log_2 N\right)$. Depending on the size of N and r, this is a relatively high communication cost. However, the larger the available bandwidth, the larger the speedup is.

6.5 Summary

In this section, applications were presented where identity-based cryptography is especially useful and where the advantage of the binding between identifier and public key is apparent. E-mail communication is not mentioned here, since the relationship between e-mail and identity-based cryptography has been discussed in the literature. The application to IP networks is especially interesting due to the global DNS system, which then turns into a public key distribution system.

The application to telephony is apparent, since the caller either has to know the number or he can use the public telephone book to retrieve it. Together with the optimization extension, the GSM scenario becomes even more practical.

7 Experimental Results

"Errors using inadequate data are much less than those using no data at all."
Charles Babbage

7.1 Introduction

This chapter presents measurements related to the entire scheme.

The presented algorithms were implemented on a standard laptop and also on a mobile device (for the particular model and configuration, see below). Measurements regarding different bit sizes of the modulus and of the involved random integers are described.

In the second part of the chapter, an efficiency comparison with the Guillou-Quisquarter signature scheme [59] (Abbr: (GQSS)) is presented, based on the number of necessary multiplications per signature. The Guillou-Quisquarter scheme is known to be efficient and the number of multiplications is the usual method to show how good a scheme performs.

Finally, the performance gain based on the the server-aided optimization algorithm described in the previous chapter is presented.

The results of this chapter were published in [122], [121] and [110] and [106].

7.2 Measurements

The algorithms were implemented in three different languages. First, a `C++` implementation using the MIRACLE arbitrary precision library [123], which can be used for academic purposes for free, was produced. Second, an implementation in `Java` by using Sun's native `BigInteger` library in the version `v1.6`, was performed. Last, `Mathematica v4.1` was used, which is a computer algebra system and supports natively many number theory functions. All measurements were done on a 2.4 GHz DualCore IBM Thinkpad with 2 GB of RAM.

7.2.1 Algorithms During Key Agreement

In Table 7.1, the time (in milliseconds) to execute the BuildSIK algorithm with different involved bit sizes is shown. The BuildSIK algorithm generates the session initiation key and is not executed during call establishment. The algorithm can be executed already in advance (i.e. *offline*) to save time during key agreement.

Alg. BuildSIK												
$\lvert r_{ID} \rvert$	$\lvert N \rvert$											
	512			**1024**			**2048**			**4096**		
	C++	Java	M	C++	Java	M	C++	Java	M	C++	Java	M
64	0.2	0.4	0.5	0.7	1.4	1.2	2.6	5.4	3.5	10.5	21.5	11.0
128	0.4	0.8	1.0	1.2	2.8	2.5	4.3	10.7	7.9	17.6	41.2	24.0
256	0.7	1.6	2.1	2.1	5.4	5.4	7.7	21.0	16.0	29.7	80.5	51.6
512	1.3	3.0	4.4	4.0	10.7	11.0	14.5	41.0	32.0	56.3	158.0	103.1

Figure 7.1: BuildSIK. Unit = ms, Rounds = 5000, G = 2 and $\lvert R \rvert = 16$

On the contrary, the Compute algorithm must be executed *online* during key agreement, since its input depends on the communication partner. It an be seen that even the most secure combination with involved bit sizes of 4096 and 512 the algorithm only takes 61.9 ms when using `C++` and 166 ms when using `Java`.

Alg. Compute												
$\lvert r_{ID} \rvert$	$\lvert N \rvert$											
	512			**1024**			**2048**			**4096**		
	C++	Java	M	C++	Java	M	C++	Java	M	C++	Java	M
64	0.3	0.6	1.1	1.0	2.2	2.7	3.5	7.8	7.5	13.7	30.7	22.4
128	0.4	1.0	1.8	1.4	3.4	4.6	5.2	13.5	12.8	20.5	50.0	40.9
256	0.7	1.8	3.1	2.3	6.0	8.7	8.7	23.8	24.8	33.3	89.7	79.4
512	1.3	3.3	6.1	4.3	11.4	16.2	15.5	45.0	49.1	61.9	166.0	154.1

Figure 7.2: Compute. Unit = ms, Rounds = 5000, G = 2 and $\lvert R \rvert = 16$

A remark should be made to the $\text{BuildSIK}_{\text{MultIDPKG}}$ and $\text{Compute}_{\text{MultIDPKG}}$ algorithms.

Essentially, these two algorithms are identical to the algorithms in the single ID-PKG case, except that they expect larger input variables. Thus, the time for the $\mathsf{Compute}_{\mathsf{MultIDPKG}}$ algorithm with two 512-bit moduli involved and 128-bit random exponents, will be equal to the time for the Compute algorithm with a single 1024-bit modulus and a 256-bit random exponent.

The Extend algorithm is used when a second ID-PKG is involved. Since no expensive exponentiation is involved, the execution time of the algorithm is the shortest of all algorithms.

Alg.: Extension												
$\lvert r_{ID} \rvert$	$\lvert N_1 \rvert = \lvert N_2 \rvert$											
	512			**1024**			**2048**			**4096**		
	`C++`	`Java`	`M`	`C++`	`Java`	`M`	`C++`	`Java`	`M`	`C++`	`Java`	`M`
N/A	0.02	1.0	1.5	0.05	3.0	3.4	0.5	10.0	8.6	1.6	42.0	24.1

Figure 7.3: Extension. Unit = ms, Rounds = 5000, $G = 2$ and $|R| = 16$

7.2.2 Algorithms for Signatures

The next two tables contain the measurements for the signature algorithms. The average time for the SigGen algorithm is shown in Table 7.4. It can be seen that the `C++` implementation gets the more efficient the larger the involved exponents are. This is probably due to the more sophisticated implementations of exponentiation algorithms in the arbitrary precision library and the known better performance of `C++` in general.

Alg. SigGen												
$\log_2 r_{ID}$	$\log_2 N$											
	512			**1024**			**2048**			**4096**		
	`C++`	`Java`	`M`	`C++`	`Java`	`M`	`C++`	`Java`	`M`	`C++`	`Java`	`M`
64	0.6	0.6	0.5	2.2	2.0	1.2	8.5	8.0	3.4	29.7	31.4	10.3
128	0.8	1.0	1.0	3.1	3.4	2.6	12.4	13.1	7.2	36.5	48.2	22.4
256	1.0	1.7	2.0	4.3	6.0	5.3	15.3	23.0	15.1	49.7	90.4	47.8
512	1.7	3.1	4.1	6.6	11.1	10.6	23.6	42.6	31.0	75.8	158.7	97.6

Figure 7.4: SigGen. Unit = ms, Rounds = 5000, $G = 2$ and $|R| = 16$

The signature verification is independent of the involved exponents and can be executed much faster than the generation algorithm.

Alg. SigVer												
$\log_2 r_{ID}$	$\log_2 N$											
	512			**1024**			**2048**			**4096**		
	C++	Java	M	C++	Java	M	C++	Java	M	C++	Java	M
N/A	0.5	0.6	0.6	1.7	1.9	1.4	6.1	7.3	3.9	24.5	28.4	11.3

Figure 7.5: SigVer. Unit = ms, Rounds = 5000, G = 2 and |R| = 16

7.2.3 Key Agreement on a Mobile Device

The basic protocol was also implemented on a state-of-the art mobile phone using `C++`. Performance measurements were made on a `Nokia N82-1` running `Symbian 9.2 FP1` with a `ARM-11 CPU` running at 330 MHz. The measurements are only related to the Compute algorithm, which is the main online computation method and determines the delay noticed by a user. For each of the 64 parameter constellations, 100 runs were performed to get a mean value. The results are presented in Figure 7.6.

512-bit modulus				
$\log_2 r_{ID}$	R			
	3	**17**	**513**	**65537**
64-bit	38	45	47	51
128-bit	86	86	82	92
256-bit	156	166	161	167
512-bit	324	335	318	325

1024-bit modulus				
$\log_2 r_{ID}$	R			
	3	**17**	**513**	**65537**
64-bit	161	174	172	180
128-bit	305	316	316	311
256-bit	620	618	629	625
512-bit	1219	1237	1240	1244

2048-bit modulus				
$\log_2 r_{ID}$	R			
	3	**17**	**513**	**65537**
64-bit	622	670	670	700
128-bit	1192	1186	1208	1169
256-bit	2320	2421	2334	2435
512-bit	4577	4559	4582	4575

4096-bit modulus				
$\log_2 r_{ID}$	R			
	3	**17**	**513**	**65537**
64-bit	2354	2485	2566	2680
128-bit	4586	4594	4734	4842
256-bit	8813	9280	9153	9100
512-bit	17641	18514	17497	17749

Figure 7.6: Mobile benchmarks. Compute. Unit = ms.

The mobile benchmarks illustrate the large difference between the execution of cryptographic operations on a desktop PC and a mobile device. On the average, the execution time is around 100 times larger compared to the desktop computer. This problem is again addressed in Chapter 6.4.

7.3 Comparison to the Guillou-Quisquater Scheme

GQSS uses the following setup. An integer $N = PQ$ with $P = 2P' + 1$ and $Q = 2Q' + 1$ with $P, Q, P', Q' \in \mathbb{P}$, with the mandatory requirement that all these primes are of cryptographic secure size. A prime ν with $\nu < \min(P, Q)$ and a one-way hash function H with $|H(m)| < \nu$ for some input m, see [58].

Public Shared Parameter of GQSS: PSP $= (N, \nu, H)$.
Secret Parameters of GQSS: SP $= (P, Q)$.

The parameters of this scheme have the same bit length as the corresponding parameters of the SSF scheme.

Definition 7.3.1 (Bit length definitions for GQSS) *Let (N, ν, H) be the public, shared parameters of the GQSS protocol with the following properties. N is a balanced RSA integer of bit length n. The bit length of the integer ν is γ. The hash function is defined by $H : \{0, 1\}^* \to \{0, 1\}^w$, thus producing a w-bit output.*

Since GQSS also makes use of a hash-function the output length was set to w-bit, to be equal Definition 5.2.1. Let ID be the identity of a participant with $|ID| < N/2$, then the identity key in the GQSS is $d_{ID} \equiv \text{Red}(ID)^{-\nu^{-1}} \pmod{N}$. The function Red() is the concatenation of ID and a redundancy depending on ID, see [59]. A signature for a message m is a triple (d, z, m) which is constructed in the following way.

7.3.1 Precomputation Optimization

Whenever the base of an exponentiation is static and known in advance, precomputation of certain integers gives a speedup in relation to the amount of precomputed data. In the presented signature algorithm the three exponentiations can be reduced to only two exponentiations, but with larger exponents. Therefore, the terms $g^{\alpha a}$ and $g^{\alpha R}$ are computed directly, rather than in a successive way. The first exponent is $\alpha \cdot h$ and the second $\alpha \cdot R$, which are of size $\approx k + w$ and $\approx k + \lambda$ respectively. The advantage is, that both exponentiations are then done to the fixed base G, which enables precomputation techniques.

Fast precomputation algorithm, like the BGMW-Algorithm [25] or the LimLee Algorithm [75] (LLA) can give a significant speed-up, depending on the amount of precom-

	Okamoto-Tanaka and SSF Gen.	GQSS Gen.	Precomp. (bit)
Without Precomp.	$\frac{3}{2}(k+w+\lambda)$	$\frac{3}{2}(w+\gamma)$	-
With Precomp .	$\frac{2^h-1}{2^h}(\left\lceil\frac{k+w}{h}\right\rceil+\left\lceil\frac{k+\lambda}{h}\right\rceil)+\left\lceil\frac{k+w}{h\nu}\right\rceil+\left\lceil\frac{k+\lambda}{h\nu}\right\rceil-4$	$\frac{2^h-1}{2^h}\left\lceil\frac{w}{h}\right\rceil+\left\lceil\frac{w}{h\nu}\right\rceil-2+\frac{3}{2}\gamma$	$(2^h-1)\nu\log_2 N$

Table 7.1: Number of Multiplications

puted data. Setting $a = \left\lceil\frac{\log_2 e}{h}\right\rceil$ and $b = \left\lceil\frac{a}{\nu}\right\rceil$ and precomputing $(2^h-1)\nu$ values, the number of multiplications of the LLA is on the average $\frac{2^h-1}{2^h}a + b - 2$. Since this value can be arbitrary small if someone decides to precompute nearly all values, the amount of precomputed data was set to a realistic value. For mobile phones, mainly small resource constrained devices are used and not much storage can be spared for precomputed data, thus the maximum amount of precomputed data is limited to 8KB. The adjacent Table 7.1 shows the theoretical number of multiplications in the average case when LLA is used for precomputation.

7.3.2 Performance Comparison

In this section, a performance comparison of the basic signature algorithm is given. Since there are no other identity-based multi-signature schemes with independent ID-PKGs, only the basic algorithm for the single signature case is compared, since for this case the similarities with the Guillou-Quisquater identity-based signature scheme [60] offers a good basis for comparison. The performance evaluation is done based on the number of multiplications needed for signature generation and verification. A brief introduction to the GQSS and the setup used for evaluation is shown to appendix 7.3. To allow for a neutral performance evaluation which is not affected by differences in implementation the analysis only consider the involved number of multiplications, which is the main factor determining the run time of the algorithm. Therefore, the exponentiation computation is broken down into the number of multiplication when using the Repeated-Squaring Method (Abbr: RSM) [25]. The average case of the RSM are $\frac{3}{2}\log_2(e)$ multiplications, when e is the given exponent. Later, the case is discussed where the precomputation is used for the case where the base of an exponentiation is known in advance. Only the multiplications in the exponentiation are taken into account, since they are the dominant factor.

Signature Generation with the Okamoto-Tanaka and SSF Protocol. In the basic algorithm for a signature generation, three exponentiations (Algorithm 15, Line 2

- Line 4) are performed, one with a k-bit exponent, one with a w-bit exponent and one with a λ-bit exponent. Using the mentioned average case of the RSM and assuming that all these integers have random characteristic there are $\frac{3}{2}(k+w+\lambda)$ multiplications on the average.

Signature Generation with the GQSS Protocol. In the GQSS algorithm for a signature generation two exponentiations (Algorithm 13, Line 2 & Line 4) are performed, one with a γ-bit exponent and one with a w-bit exponent. Using the mentioned average case of the RSM and assuming that all these integers have random characteristic there are $\frac{3}{2}(w+\gamma)$ multiplications on the average.

In direct comparison this gives the Guillou-Quisquater signature scheme an advantage of $\frac{3}{2}k$ multiplications. If a user decides to leave the random k-bit integer α in the presented Okamoto-Tanaka and SSF signature algorithm random but fixed, the runtime of both algorithm are equal. Furthermore, in both protocols the integer R and ν can be chosen to contain only few 1s in its binary representation.

Signature Verification in the Okamoto-Tanaka and SSF Protocol. In the algorithm for a signature verification two exponentiations (Algorithm 16, Line 3 & Line 4) are performed, one with a λ-bit exponent and one with a w-bit exponent. Using the mentioned average case of the RSM and assuming that all the integers have random characteristic there are $\frac{3}{2}(w+\lambda)$ multiplications on the average.

Signature Verification in the GQSS Protocol. In the GQSS algorithm for a signature verification two exponentiations (Algorithm 14, both Line 2) are performed, one with a γ-bit exponent and one with a w-bit exponent. Using the mentioned average case of the RSM and assuming that all the integers have random characteristic there are $\frac{3}{2}(w+\gamma)$ multiplications on the average.

Since λ and γ can be chosen freely, the runtime of the two algorithms are equal.

However, since the GQSS protocol uses the random integer r in the base (Algorithm 13, Line 2), rather than an exponent, like the proposed approach does, the precomputation speedup presented in appendix 7.3.1 can not be applied to the GQSS protocol. Thus, the presented protocol can outperform the GQSS protocol. In Table 7.2 example values for different exponent sizes are shown for signature generation. The modulus is a 2048-bit RSA integer, and the precomputation is determined by $\nu = 4$ and $h = 3$. The precomputed data is within the 8KB limit. The value k is the bit size of an exponent which need to be adequately large to prevent guessing of the exponent. It can vary

bit sizes	OkTa/SSF Gen.	GQSS Gen.	Precomp. (KB)
$k = 64$, $w = 256$, $\lambda = \gamma = 256$	237	479	7.198
$k = 128$, $w = 256$, $\lambda = \gamma = 256$	288	479	7.198
$k = 256$, $w = 256$, $\lambda = \gamma = 256$	381	479	7.198
$k = 64$, $w = 256$, $\lambda = \gamma = 512$	334	863	7.198
$k = 128$, $w = 256$, $\lambda = \gamma = 512$	383	863	7.198
$k = 256$, $w = 256$, $\lambda = \gamma = 512$	478	863	7.198
$k = 64$, $w = 256$, $\lambda = \gamma = 1024$	525	1631	7.198
$k = 128$, $w = 256$, $\lambda = \gamma = 1024$	576	1631	7.198
$k = 256$, $w = 256$, $\lambda = \gamma = 1024$	669	1631	7.198

Table 7.2: Number of Multiplications

between 2^{64} and 2^{256}. It is $w = 256$ since SHA-256 is a well known and often utilized hash function that has 256 bit. Since v must be larger than $\mathsf{H}(m)$ it must hold $\gamma > w$. And since λ and γ are equivalent in the two protocols it is $\lambda = \gamma$. As can be seen in Table 7.2, the SSF algorithm requires significantly less multiplications than the GQSS scheme.

7.4 Server-Aided Optimization

To estimate how much the outsourcing algorithm improves the performance of the overall execution time of the encryption process, the measurements of Table 7.6 in Chapter 7 have to be considered. The figure shows the time to compute the encryption key. Obviously, the algorithm lasts longer, the larger the involved bit sizes are. To illustrate the performance gain, we pick a 4096-bit modulus and choose different exponent sizes: 64-bit,128-bit,256-bit,512-bit and 1024-bit. The time for a computation for these sizes on the mobile phone is around 2.4 seconds, 4.7 seconds, 8.9 seconds, 17 seconds, and 34.7 seconds, respectively. Thus, if the total transfer time to the backend (plus the computation time on the backend $\mathcal{B}$, which is mostly negligible compared to the other terms) is less than these values in seconds, a theoretical speedup is achieved. However, the actual bandwidth and connection properties can have fluctuations, caused by startup problems and jitter. In the latter case, this unknown factor is covered by an additional term called t_{delay}.

In all measurements, R is neglected, since it does not change the time significantly. The theoretical performance gain is due to the reduction of the execution time on the mobile phone by the corresponding total transfer time. The plain transfer times are shown in Figure 7.7, whereas the gained speedup in seconds is shown in Figure 7.8.

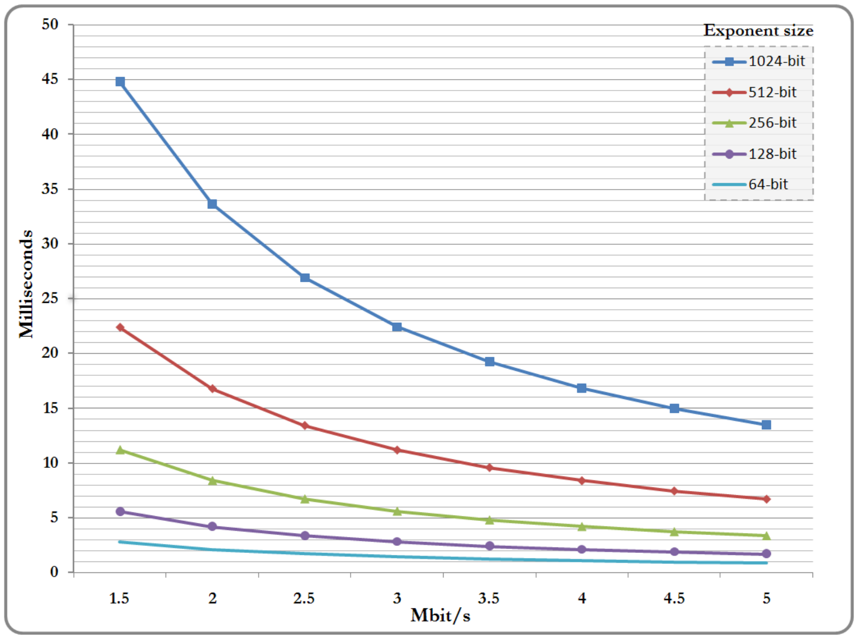


Figure 7.7: Time (in seconds) to transfer all necessary bits for a 4096-bit modulus.

Example: Take the measurements of Chapter 7 Table 7.2 as a minimum speed-up. The actual speedup when using a cluster node will be probably much higher. Suppose there is a 3.5 Mbit/s connection and a 4096-bit modulus together with a 256-bit random exponent. The time for the Compute algorithm based this combination on the Nokia telephone is around 9.1 seconds. With a look at the Figure 7.7, it can be seen that 5 seconds are necessary for the transport of all arising bits. Thus, computation time reduces from around 9 seconds to $\approx$ 5 seconds (C++ implementation).

It is evident that with a 1.5 Mbit/s connection, no speedup can be achieved. However, already with a 2 Mbit/s connection, all differences are positive, which will probably be counterbalanced by startup times and jitter. If the bandwidth increases further, the obtained seconds become significant for the larger exponent sizes. To reduce the waiting times for the phone call to be established by several seconds, is a quite noticeable improvement for both callee and caller.

After having illustrated the theoretical speedup, the value $\mathsf{t}_{\mathsf{delay}}$ will now be increased. To uncover its effect, it is picked for a 4096-bit modulus and a 512-bit exponent. To total amount of data that must be transferred in this constellation is, according to Eq. 6.5, around 4.2 MBytes. The communication between the mobile device and the server is a normal data connection with only a single connection establishment. Two computations

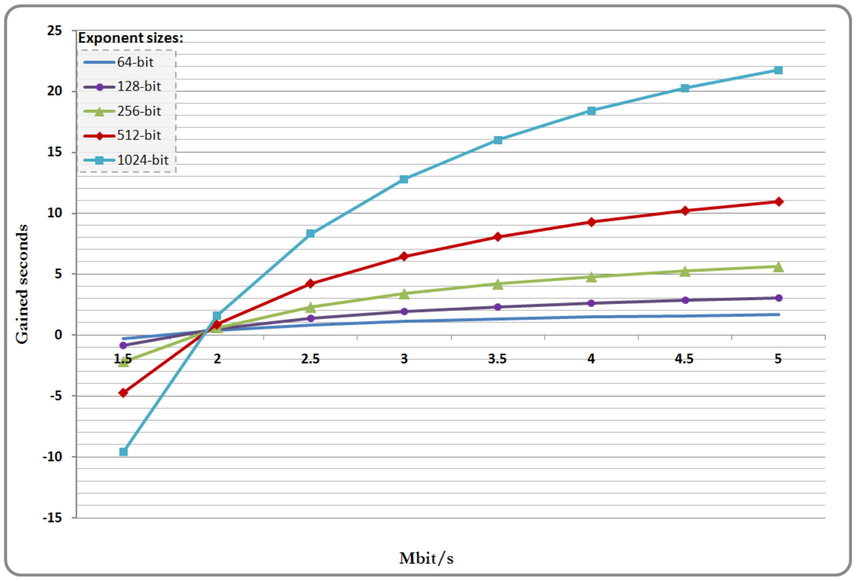


Figure 7.8: Time gained (in seconds) based on the outsourcing method for a 4096-bit modulus regarding different exponent sizes.

on the corresponding sites can be done on time, so no new connection must be built up in each round. The theoretical transfer time of these 4.2 MBytes can now differ regarding the value t_{delay}. Based on this fact, it is clear that this algorithm obtains a speedup for computations in local rea networks or other high speed environments, since there the time to transfer a few MBytes is negligible. In a GSM environment, the average jitter time is per default set to 4 ms. This is the value that is given in the literature and is used in many specifications. This means that the average transfer time of a packet is, on the average, delayed by 4 ms. Additionally, we have a startup delay that also decreases the performance but only occurs once. The average of this startup delay is around 300 ms in the normal GSM network and below 50 ms in modern environments like HSPA. However, this constitutes only an additional summand and does not influence the performance as much as jitter does. In Figure 7.9, the jitter parameter is varied from 0 to 2 times on the average, thus 8 ms per communication. The startup delay is set to 300 ms.

High jitter and a low bandwidth leeds to a slowdown characterized by the upper right negative table entries. But whenever the bandwidth reaches the 3 Mbit/s level (UMTS speed), even a network with a 2 times average jitter gives a slight speedup. In a 5 Mbit/s network with a 2 times average jitter, we gain 7 seconds compared to the case when all computations are performed on the mobile phone. For comparison, the speedup in local area network environments is also displayed.

Mbits/s	jitter								
	0 ms	**1 ms**	**2 ms**	**3 ms**	**4 ms**	**5 ms**	**6 ms**	**7 ms**	**8 ms**
1.5	-4.35	-5.55	2566	-7.59	-8.61	-9.64	-10.66	-11.68	-12.70
2	1.06	0.03	-0.99	-2.01	-3.03	-4.05	-5.08	-6.10	-7.12
2.5	4.40	3.38	2.36	1.34	0.32	-0.71	-1.73	-2.75	-3.77
3	6.64	5.62	4.59	3.57	2.55	1.53	0.51	-0.52	-1.54
3.5	8.23	7.21	6.19	5.17	4.14	3.12	2.10	1.08	0.06
4	9.43	8.41	7.38	6.36	5.34	4.32	3.30	2.27	1.25
4.5	10.36	9.34	8.31	7.29	6.27	5.25	4.23	3.20	2.18
5	11.10	10.08	9.06	8.04	7.01	5.99	4.97	3.95	2.93
10	14.45	13.43	12.41	11.39	10.36	9.34	8.32	7.30	6.28
100	17.47	16.44	15.42	14.40	13.38	12.36	11.33	10.31	9.29
1000	17.49	16.74	15.72	14.70	13.68	12.66	11.63	10.61	9.59

Figure 7.9: Startup = 300 ms. The tables shows the gained speedup in seconds. That means the 17.5 seconds from the mobile computation must be reduced by the value from the table. E.g. for a 3 Mbit/s line and 2 ms jitter, the 17.5 seconds compute time is reduced by 4.56 seconds.

Obviously, the algorithm is not optimal for GSM networks, due to its its jitter and startup times. However, it can lead to a speedup, at least at the UTMS speed level and decreases the time a user has to wait to get the encrypted phone call established. Furthermore, it is the first algorithm that allows to outsource an exponentiation where the base is unknown and the exponent must be kept secret.

7.5 Summary

The measurements show that the proposed scheme performs well in general and in comparison with the Guillou-Quisquater scheme. Even in the case of 4096-bit moduli, the C++ implementation only needs around 62 ms to compute the session key, which illustrates the time a user is delayed based on the key agreement process. The time for the Extension algorithm, which is executed once when using two ID-PKGs, can be neglected. The signature scheme performs well, too. It takes only 76 ms to generate a signature using a 4096-bit modulus as well as a 512-bit exponent. The verification is done in only 25 ms and is independent of the random exponent. In the contrast to GQSS, the SSF scheme can benefit from precomputation, which allows to outperform the GQSS. The analysis of the server-aided computation extension shows that the algorithm performs a speedup whenever the available bandwidth is at least 3Mbit/s.

8 Conclusions

"A conclusion is the place where you got tired of thinking."
Arthur Bloch

8.1 Summary

The main contribution of this thesis is the development of the Secure Session Framework. It consists of two main parts: first, a key agreement scheme with extensions to multiple independent key generators, and second, a corresponding multi-signature scheme. The key agreement is based on well known assumptions and is efficient in the terms of communication and computational cost. It fulfills all necessary requirements for a *secure and authenticated key agreement protocol*, which was proven using the Canetti-Krawczyk Model [30, 31]. For the multiple ID-PKG case, the proof given by Gennaro et al. [48] was followed. The signature scheme was proven secure against *existential forgery on adaptively chosen message and ID attacks.* This property was separately proven for all three introduced versions: single signature with single ID-PKG, multi-signature with single ID-PKG and multi-signatures with multiple ID-PKGs. Thus, with the proposed way to handle independent ID-PKGs, an open problem in the field of IBC was solved.

The next part of the thesis was about related attacks. The Φ-Hiding assumption was addressed and it was shown that this assumption can be broken with an average advantage probability of 1/4 if the setup was chosen in a certain way. This is a quite surprising result, since the Φ-Hiding assumption is deeply associated with the Integer Factorization Problem, which again has not been solved for centuries. In the general case (using composite integers to hide, rather than primes), the invented attack even gets more powerful. This means that the probability to break the assumption increases towards 1/2, the more prime factors the hidden integers contain. The second attack was directed to secret sharing schemes. In this case, the findings were as follows: Whenever a CRT-based threshold secret sharing scheme is used to distribute the integer $\varphi(N)$ (in SSF: the master secret key) among several ID-PKGs, a subset of malicious ID-PKGs can reveal the entire secret under certain circumstances using lattice based reduction methods. Some publications, e.g. the one of Iftene and Grindei [65], indeed use this kind of setup and could be shown to be insecure.

The last part of the thesis presented applications and experimental results. Its focus was on two scenarios, IPv4 networks and GSM/VoIP communication. For the first scenario, real world issues were discussed and problems an adopter has to deal with. Dynamic IP addresses, NAT traversal and secure distribution of the involved keys are some of these obstacles. Additionally, it was illustrated how the SSF signature scheme can be used to prevent IP spoofing by signing a timestamp as a proof of possession. The second scenario was about GSM and VoIP communication. Because the actual encryption used for GSM communication is insecure, the need for an end-to-end encryption method is apparent. Therefore, it was shown how SSF can be built into a GSM architecture, and a prototype implementation on a `Nokia N95` was performed. Regarding VoIP, several implementations were made to show the practicability. The open source implementations Jabbin and WengoPhone were extended with the SSF protocol and corresponding SIP registrars were enhanced to generate the identity keys for each user. Because of the limited computational power of mobile devices, an optimization was presented by using server-aided cryptography to outsource expensive computation to powerful backends.

8.2 Future Work

There are several open problems that need to be addressed:

Group Key Agreement / Group Signature. A key agreement is defined as an action that takes place between two entities. For an application like VoIP or GSM telephony it perfectly makes sense to utilizes this concept, since most of the time only two participants communicate.

Based on further developments of Internet technology, by now participants tend to use more and more conference conversations that allow to share a communication channel between several entities. For a conference with n attendees, it would be possible to make a pairwise key agreement and to encrypt a message with $n - 1$ different keys. However, this is an unnecessary overhead and could be reduced when using so called *group key agreements*. Figure 8.1 illustrates the number of messages for both cases, the pairwise key agreement and a normal group key agreement scheme. In the latter, each participant sends a message to its left neighbor and the last receiver broadcasts the final packet to each previous participant.

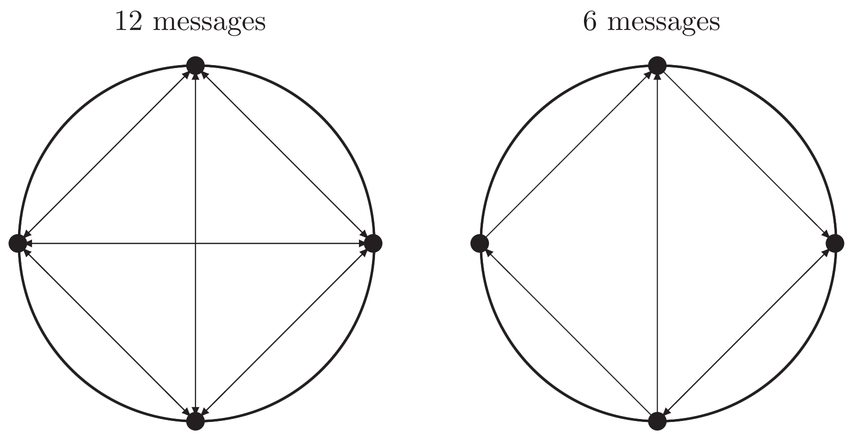


Figure 8.1: The number of arrowheads indicate the number of messages involved. On the left side, which shows pairwise key agreements, $4(4-1) = 12$ messages are necessary; on the right side, only 6 message are required.

Thus, an extension to the SSF scheme to allow group key agreements in a secure way, by using similar ideas as Steiner et al. [124], is an interesting area of future work. Since GSM, VoIP and even chat are perfect applications for IBC, conference conversation is the next logical step.

Elliptic Curves. Elliptic curves have the advantage that they achieve the same level of security as schemes that are based on classical assumptions, but require less bits. Consequently, they are more efficient regarding bandwidth consumption. Since the CRT can also be used to characterize points on elliptic curves, the idea to make a completely independent ID-PKG work can perhaps be transferred to elliptic curves as well.

Implementations. The implementation of SSF was done on various platforms and devices. However, some of them were not completely elaborated. Using the data channel of a mobile phone is the straightforward way for the implementation on a mobile phone. Even if the data channel has a very low bandwidth, it is very comfortable to use since it offers reliable data transfer. But since the providers are going to narrow the support for data channels and some even do not support them any more, one has to switch from the data channel to the voice channel. This entails the problem that the voice channel is subject to data compression. If a packet that contains encrypted data is transferred and loses bits due to a compression routine, the decryption process will fail. To make this implementation work, ideas as those described by LaDue et al. [72] are good examples of how such a implementation should and could be done in the future.

Bibliography

[1] Adida, B., Chau, D., Hohenberger, S., and Rivest, R. L. Lightweight Email Signatures. In *SCN'06 - Proceedings of 5th International Conference on Security and Cryptography for Networks* (2006, Maiori, Italy), vol. 4116 of *Lecture Notes in Computer Science*, Springer, pp. 288–302.

[2] Agel, B. Sichere Schlüsseleinigung im GSM-Mobilfunknetz. Master's thesis, Philipps-University of Marburg, 2008.

[3] Aggarwal, D., and Maurer, U. Breaking RSA Generically is Equivalent to Factoring. In *EUROCRYPT - Advances in Cryptology* (2009, Cologne, Germany), vol. 5479 of *Lecture Notes in Computer Science*, Springer, pp. 36–53.

[4] Al-Riyami, S., and Paterson, K. Tripartite Authenticated Key Agreement Protocols from Pairings. In *Proceedings of the 9th IMA International Conference on Cryptography and Coding* (2003, Cirencester, UK), vol. 2898 of *Lecture Notes in Computer Science*, pp. 332–359.

[5] Andersen, D. G., Balakrishnan, H., Feamster, N., Koponen, T., Moon, D., and Shenker, S. Holding the Internet Accountable. In *Proceedings of the 6th ACM Workshop on Hot Topics in Networking (Hotnets)* (November 2007, Atlanta, GA), pp. 51–54.

[6] Asmuth, C., and Bloom, J. A Modular Approach to Key Safeguarding. *IEEE Transactions on Information Theory 29*, 2 (1983), 208–210.

[7] Aura, T. Cryptographically Generated Addresses, 2005. RFC 3972.

[8] Baker, F. Requirements for IP Version 4 Routers, 1995. RFC 1812.

[9] Bellare, M., Canetti, R., and Krawczyk, H. Message Authentication Using Hash Functions: the HMAC Construction. *CryptoBytes 2*, 1 (Spring 1996),

12–15.

[10] Bellare, M., and Neven, G. Multi-Signatures in the plain Public-Key Model and a General Forking Lemma. In *CCS'06 - Proceedings of the 13th ACM Conference on Computer and Communications Security* (2006, Alexandria, Virginia, USA), ACM Press, pp. 390–399.

[11] Bellare, M., and Neven, G. Identity-Based Multi-signatures from RSA. In *Topics in Cryptology - CT-RSA 2007, The Cryptographers' Track at the RSA Conference* (2007, San Francisco, CA, USA), vol. 4377 of *Lecture Notes on Computer Science*, pp. 145–162.

[12] Bellare, M., and Rogaway, P. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. In *CCS'93 - Proceedings of the 1st ACM Conference on Computer and Communications Security* (1993, Fairfax, Virginia, United States), ACM Press, pp. 62–73.

[13] Blakely, G. Safeguarding Cryptographic Keys. In *Proceedings of the National Computer Conference* (1979), vol. 48, pp. 313–317.

[14] Bloemer, J., and May, A. New Partial Key Exposure Attacks on RSA. In *CRYPTO - Advances in Cryptology* (2003, Santa Barbara, California, USA), vol. 2729 of *Lecture Notes in Computer Science*, Springer, pp. 27–43.

[15] Boneh, D. The Decision Diffie-Hellman Problem. In *Algorithmic Number Theory* (1998), vol. 1423 of *Lecture Notes in Computer Science*, pp. 48–63.

[16] Boneh, D., Boyen, X., and Goh, E.-J. Hierarchical Identity Based Encryption with Constant Size Ciphertext . In *EUROCRYPT - Advances in Cryptology* (2005, Aarhus, Denmark), vol. 3494 of *Lecture Notes in Computer Science*, Springer, pp. 440–456.

[17] Boneh, D., and Durfee, G. Cryptanalysis of RSA with private key d less than $\mathsf{N}^{0.292}$. *IEEE Transactions on Information Theory 46* (1999), 1339–1349.

[18] Boneh, D., Durfee, G., and Frankel, Y. An Attack on RSA Given a Small Fraction of the Private Key Bits. In *ASIACRYPT - Advances in Cryptology* (1998, Beijing, China), vol. 1514 of *Lecture Notes in Computer Science*, Springer, pp. 25–34.

[19] Boneh, D., and Franklin, M. Identity-Based Encryption from the Weil Pairing. *SIAM Journal of Computation 32*, 3 (2003), 586–615.

[20] Boneh, D., Gentry, C., and Hamburg, M. Space-Efficient Identity-based Encryption Without Pairings. In *FOCS'07 - Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science* (2007, Rhode, Island), IEEE Computer Society, pp. 647–657.

[21] Boneh, D., and Hamburg, M. Generalized Identity-based and Broadcast Encryption Systems Motivated by Secure Email. In *ASIACRYPT - Advances in Cryptology* (2008, Melbourne, Australia), vol. 5350 of *Lecture Notes in Computer Science*, Springer, pp. 344–354.

[22] Boneh, D., and Shacham, H. Fast Variants of RSA. *CryptoBytes 5*, 1 (Winter/Spring 2002), 1–9.

[23] Boneh, D., and Venkatesan, R. Breaking RSA may not be equivalent to Factoring. In *EUROCRYPT - Advances in Cryptology* (1998, Espoo, Finland), vol. 1403 of *Lecture Notes in Computer Science*, Springer Berlin, pp. 59–71.

[24] Boyen, X., and Waters, B. Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In *CRYPTO - Advances in Cryptology* (2006, Santa Barbara, California, USA), vol. 4117 of *Lecture Notes in Computer Science*, Springer, pp. 290–307.

[25] Brickell, E. F., Gordon, D. M., McCurley, K. S., and Wilson, D. B. Fast Exponentiation with Precomputation: Algorithms and Lower Bounds. In *EUROCRYPT - Advances in Cryptology,* (1992, Balatonfured, Hungary), vol. 658 of *Lecture Notes in Computer Science*, Springer Berlin, pp. 200–207.

[26] Cachin, C. Efficient Private Bidding and Auctions with an Oblivious Third Party. In *CCS'99 - Proceedings of the 6th ACM Conference on Computer and Communications Security* (1999, Kent Ridge Digital Labs, Singapore), ACM Press, pp. 120–127.

[27] Cachin, C., Micali, S., and Stadler, M. Computationally Private Information Retrieval with Polylogarithmic Communication. In *EUROCRYPT - Advances in Cryptology* (1999, Prague, Czech Republic), vol. 1592 of *Lecture Notes in Computer Science*, Springer, pp. 402–407.

[28] Canetti, R., Goldreich, O., and Halevi, S. The Random Oracle Methodology, Revisited. *Journal of the ACM 51*, 4 (2004), 557–594.

[29] Canetti, R., Halevi, S., and Katz, J. A Forward-Secure Public-Key Encryption Scheme. In *EUROCRYPT - Advances in Cryptology* (2003, Warsaw, Poland), vol. 2656 of *Lecture Notes in Computer Science*, Springer, pp. 255–271.

[30] Canetti, R., and Krawczyk, H. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In *EUROCRYPT - Advances in Cryptology* (2001, Innsbruck, Austria), vol. 2045 of *Lecture Notes in Computer Science*, pp. 453–474.

[31] Canetti, R., and Krawczyk, H. Universally Composable Notions of Key Exchange and Secure Channels. In *EUROCRYPT - Advances in Cryptology* (2002, Amsterdam, The Netherlands), vol. 2332 of *Lecture Notes in Computer Science*, pp. 337–351.

[32] Chen, L., Cheng, Z., and Smart, N. P. Identity-based Key Agreement Protocols from Pairings. *International Journal of Information Security 6*, 4 (2007), 213–241.

[33] Cheng, X., Liu, J., and Wang, X. Identity-based Aggregate and Verifiably Encrypted Signatures from Bilinear Pairing. In *International Conference on Computational Science and Its Application* (2005, Singapore), pp. 1046–1054.

[34] Clavier, C. An Improved SCARE Cryptanalysis Against a Secret A3/A8 GSM Algorithm. In *3th International Conference on Information Systems Security* (2007, Delhi, India), pp. 143–155.

[35] Cocks, C. An Identity-based Encryption Scheme Based on Quadratic Residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding* (2001, Cirencester, UK), vol. 2260 of *Lecture Notes in Computer Science*, Springer, pp. 360–363.

[36] Congdon, P., Aboba, B., Smith, A., Zorn, G., and Roese, J. IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines, September 2003. RFC 3580.

[37] Coppersmith, D. Small Solutions to Polynomial Equations and low Exponent

RSA Vulnerabilities. In *Journal of Cryptology* (1997), vol. 10, Springer, pp. 233–260.

[38] Coron, J.-S. Finding Small Roots of Bivariate Integer Polynomial Equations Revisited. In *EUROCRYPT - Advances in Cryptology* (2004, Interlaken, Switzerland), vol. 3027 of *Lecture Notes in Computer Science*, Springer, pp. 492–505.

[39] Cramer, R., and Shoup, V. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *CRYPTO - Advances in Cryptology* (1998, Santa Barbara, California, USA), vol. 1462 of *Lecture Notes in Computer Science*, Springer, pp. 13–25.

[40] Diffie, W., and Hellman, M. E. New Directions In Cryptography. *IEEE Transactions On Information Theory*, 6 (1976), 644–654.

[41] Dijk, M., Clarke, D., Gassend, B., Suh, G. E., and Devadas, S. Speeding up Exponentiation using an Untrusted Computational Resource. *Design Codes and Cryptography 39*, 2 (2006), 253–273.

[42] Eikenberry, S. M., and Sorenson, J. P. Efficient Algorithms for Computing the Jacobi Symbol. *Journal of Symbolic Computation 26*, 4 (1998), 509–523.

[43] ElGamal, T. A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *CRYPTO - Advances in Cryptology* (1984, Santa Barbara, California, USA), vol. 196 of *Lecture Notes in Computer Science*, Springer, pp. 10–18.

[44] Ellison, C., and Schneier, B. Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure. In *Computer Security Journal* (2000), pp. 1–7.

[45] Ernst, M., Jochemsz, E., May, A., and de Weger, B. Partial Key Exposure Attacks on RSA up to Full Size Exponents. In *EUROCRYPT - Advances in Cryptology* (2005, Aarhus, Denmark), vol. 1592 of *Lecture Notes in Computer Science*, Springer, pp. 371–386.

[46] Ferguson, P., and Senie, D. Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing, 2000. RFC 2827.

[47] Gennaro, R., Halevi, S., and Rabin, T. Secure Hash-and-Sign Signa-

ture without the Random Oracle. In *EUROCRYPT - Advances in Cryptology* (1999, Prague, Czech Republic), vol. 1592 of *Lecture Notes in Computer Science*, pp. 123–139.

[48] Gennaro, R., Krawczyk, H., and Rabin, T. Okamoto-Tanaka Revisited: Fully Authenticated Diffie-Hellman with Minimal Overhead. In *ACNS - 8th International Conference on Applied Cryptography and Network Security* (2010, Beijing, China), Lecture Notes in Computer Science, p. (to appear).

[49] Gentry, C., Mackenzie, P., and Ramzan, Z. Password Authenticated Key Exchange Using Hidden Smooth Subgroups. In *CCS'05 - Proceedings of the 12th ACM Conference on Computer and Communications Security* (2005, Alexandria, VA, USA), ACM Press, pp. 299–309.

[50] Gentry, C., and Ramzan, Z. Single-Database Private Information Retrieval with Constant Communication Rate. In *ICALP'05 - Proceedings of the 32nd International Colloquium on Automata, Languages and Programming* (2005, Lisbon, Portugal), pp. 803–815.

[51] Gentry, C., and Ramzan, Z. Identity-Based Aggregate Signatures. In *Public Key Cryptography* (2006, New York, USA), pp. 257–273.

[52] Gentry, C., and Silverberg, A. Hierarchical ID-Based Cryptography. In *ASIACRYPT - Advances in Cryptology* (2002, Queenstown, New Zealand), vol. 2501 of *Lecture Notes in Computer Science*, Springer, pp. 548–566.

[53] Goldwasser, S., and Micali, S. Probabilistic Encryption & how to play Mental Poker keeping Secret all partial Information. In *STOC'82 - Proceedings of the 14th ACM Symposium on Theory of Computing* (1982, San Francisco, California, United States), ACM Press, pp. 365–377.

[54] Gordon, D. M. A Survey of Fast Exponentiation Methods. *Journal of Algorithms 27*, 1 (1998), 129–146.

[55] Goyal, V. Reducing Trust in the PKG in Identity-based Cryptosystems. In *CRYPTO - Advances in Cryptology* (2007, Santa Barbara, California, USA), vol. 4622 of *Lecture Notes in Computer Science*, Springer, pp. 430–447.

[56] Graf, T. Verschlüsselung von Datenströmen in Videokonferenzen. Master's

thesis, Philipps-University of Marburg, 2008.

[57] Guenther, C. G. An Identity-based Key-Exchange Protocol. In *EUROCRYPT - Advances in Cryptology* (1990, Houthalen, Belgium), vol. 434 of *Lecture Notes in Computer Science*, Springer, pp. 29–37.

[58] Guilin Wang and Bo Zhu. Remarks on Saeednia's Identity-Based Society Oriented Signature Scheme with Anonymous Signers. Cryptology ePrint Archive 2003/46, March 2003.

[59] Guillou, L. C., and Quisquater, J.-J. Efficient Digital Public-Key Signature with Shadow. In *CRYPTO - Advances in Cryptology* (1988, Santa Barbara, California, USA), vol. 293 of *Lecture Notes in Computer Science*, Springer, p. 223.

[60] Guillou, L. C., and Quisquater, J.-J. A Paradoxical Identity-Based Signature Scheme Resulting from Zero-Knowledge. In *CRYPTO - Advances in Cryptology* (1990, Santa Barbara, California, USA), vol. 403 of *Lecture Notes in Computer Science*, Springer, pp. 216–231.

[61] Hemenway, B., and Ostrovsky, R. Public Key Encryption which is Simultaneously a Locally-Decodable Error-Correcting Code. In *Electronic Colloquium on Computational Complexity, Report No. 21* (2007).

[62] Heng, S.-H., and Kurosawa, K. k-Resilient Identity-based Encryption in the Standard Model. In *Topics in Cryptology - CT-RSA 2004, The Cryptographers' Track at the RSA Conference* (2004, San Francisco, CA, USA), vol. 2964 of *Lecture Notes on Computer Science*, Springer, pp. 67–80.

[63] Hohenberger, S., and Lysyanskaya, A. How to Securely Outsource Cryptographic Computations. In *TCC'05 - Proceeding of the 2nd International Conference on Theory of Cryptography* (2005, Cambridge, MA, USA), vol. 3378 of *Lecture Notes in Computer Science*, Springer, pp. 264–282.

[64] Horwitz, J., and Lynn, B. Toward Hierarchical Identity-based Encryption. In *EUROCRYPT - Advances in Cryptology* (2002, Amsterdam, Netherland), vol. 2332 of *Lecture Notes in Computer Science*, Springer, pp. 466–481.

[65] Iftene, S., and Grindei, M. Weighted Threshold RSA Based on the Chinese Remainder Theorem. In *SYNASC'07 - Proceedings of the Ninth Interna-*

tional Symposium on Symbolic and Numeric Algorithms for Scientific Computing (Washington, DC, USA, 2007, Timisoara, Romania), IEEE Computer Society, pp. 175–181.

[66] Joux, A., Naccache, D., and Thomé, E. When e-th Roots become easier than Factoring. In *ASIACRYPT - Advances in Cryptology* (2007, Kuching, Sarawak, Malaysia), vol. 4833 of *Lecture Notes in Computer Science*, Springer, pp. 13–28.

[67] Kaya, K., and Selcuk, A. A. Threshold Cryptography based on Asmuth-Bloom Secret Sharing. *Information Sciences 177*, 19 (2007), 4148–4160.

[68] Kaya, K., and Selcuk, A. A. Robust Threshold Schemes Based on the Chinese Remainder Theorem. In *AFRICACRYPT 2008 First International Conference on Cryptology in Africa* (2008, Casablanca, Morocco), vol. 5023 of *Lecture Notes in Computer Science*, Springer, pp. 94–108.

[69] Kent, S., and Atkinson, R. Security Architecture for the Internet Protocol, 1998. RFC 2401.

[70] Koblitz, N. *A Course in Number Theory and Cryptography.* Springer, 1994.

[71] Kunihiro, N., Abe, W., and Ohta, K. Maurer-Yacobi ID-Based Key Distribution Revisited. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E89-A*, 5 (2006), 1421–1424.

[72] LaDue, C. K., Sapozhnykov, V. W., and Fienberg, K. A Data Modem for GSM Voice Channel. *IEEE Transactions on Vehicular Technology 57*, 4 (2008), 2205–2218.

[73] Lenstra, A. K., H. W. Lenstra, J., and Lovász, L. Factoring Polynomials with Rational Coefficients. *Mathematische Annalen 261* (1982), 515–534.

[74] Lenstra, A. K., and Hendrik W. Lenstra, J., Eds. *The Development of the Number Field Sieve*, vol. 1554 of *Lecture Notes in Mathematics.* Springer, Berlin, 1993.

[75] Lim, C. H., and Lee, P. J. More Flexible Exponentiation with Precomputation. In *CRYPTO - Advances in Cryptology* (1994, Santa Barbara, California, USA), vol. 839 of *Lecture Notes in Computer Science*, pp. 95–107.

[76] Lim, C. H., and Lee, P. J. Security and Performance of Server-Aided RSA Computation Protocols. In *CRYPTO - Advances in Cryptology* (1995, Santa Barbara, California, USA), vol. 963 of *Lecture Notes in Computer Science*, Springer, pp. 70–83.

[77] Lim, C. H., and Lee, P. J. Authenticated Session Keys and Their Server-Aided Computation, 2006. Tech. Report.

[78] Liu, X., Li, A., Yang, X., and Wetherall, D. Passport: Secure and Adoptable Source Authentication. In *USENIX/ACM Symposium on Networked Systems Design and Implementation* (Berkeley, CA, USA, 2008), USENIX Association, pp. 365–378.

[79] Liu, X., Yang, X., Wetherall, D., and Anderson, T. Efficient and Secure Source Authentication with Packet Passports. In *Proceedings of the 2nd Conference on Steps to Reducing Unwanted Traffic on the Internet* (Berkeley, CA, USA, 2006), USENIX Association, pp. 2–9.

[80] Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., and Waters, B. Sequential Aggregate Signatures and Multi-Signatures without Random Oracles. In *EUROCRYPT - Advances in Cryptology* (2006, Saint Petersburg, Russia), vol. 4004 of *Lecture Notes on Computer Science*, Springer, pp. 465–485.

[81] Matsumoto, T., Kato, K., and Imai, H. Speeding Up Secret Computations with Insecure Auxiliary Devices. In *CRYPTO - Advances in Cryptology* (1990, Santa Barbara, California, USA), vol. 537 of *Lecture Notes in Computer Science*, Springer, pp. 497–506.

[82] Maurer, U. Towards the Equivalence of Breaking the Diffie-Hellman Protocol and Computing Discrete Logarithms. In *CRYPTO - Advances in Cryptology* (1994, Santa Barbara, California, USA), vol. 839 of *Lecture Notes in Computer Science*, Springer, pp. 271–281.

[83] Maurer, U., and Yacobi, Y. A Non-interactive Public-Key Distribution System. *Design, Codes and Cryptography 9*, 3 (1996), 305–316.

[84] McCullagh, N., and Barreto, P. A New Two-Party Identity-Based Authenticated Key Agreement. In *Topics in Cryptology - CT-RSA 2007, The Cryptographers' Track at the RSA Conference, Lecture Notes on Computer Science*

(2005, San Francisco CA), vol. 3376, pp. 262–274.

[85] Menezes, A., Vanstone, S., and Okamoto, T. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. In *STOC'91 - Proceedings of the 23th Annual ACM Symposium on Theory of Computing* (1991, New Orleans, Louisiana, United States), ACM, pp. 80–89.

[86] Mignotte, M. How to Share a Secret? In *EUROCRYPT - Workshop on Advances in Cryptology* (1982, Burg Feuerstein, Germany), vol. 149 of *Lecture Notes in Computer Science*, Springer, pp. 371–375.

[87] Mirkovic, J., and Reiher, P. A Taxonomy of DDoS Attack and DDoS Defense Mmchanisms. In *SIGCOMM'04 - Proceedings of the Annual Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication* (New York, NY, USA, 2004, Portland, Oregon), ACM Press, pp. 39–53.

[88] Moskowitz, R., Nikander, P., Jokela, P., and Henderson, T. Host Identity Protocol, October 2003. RFC 4423.

[89] Nguyen, K. Index Calculus. In *Encyclopedia of Cryptography and Security.* 2005.

[90] Nguyen, P., Shparlinski, I., and Stern, J. Distribution of Modular Sums and the Security of Server Aided Exponentiation. In *In Proceedings of the Workshop on Computational Number Theory and Cryptography* (1999, Singapore), pp. 1–16.

[91] Ohta, K., and Okamoto, T. A Digital Multisignature Scheme Based on the Fiat-Shamir Scheme. In *ASIACRYPT - Advances in Cryptology* (1991, Fujiyoshida, Japan), vol. 739 of *Lecture Notes on Computer Science*, Springer, pp. 139–148.

[92] Okamoto, E. Key Distribution Systems Based on Identification Information. In *CRYPTO - Advances in Cryptology* (1987, Santa Barbara, California, USA), vol. 293 of *Lecture Notes in Computer Science*, Springer, pp. 194–202.

[93] Okamoto, E., and Tanaka, T. Key Distribution System based on Identification Information. *IEEE Journal on Selected Areas in Communications 7*, 4 (1989), 481–485.

[94] Paillier, P. Public key Cryptosystems Based on Composite Degree Residuosity Classes. In *EUROCRYPT - Advances in Cryptology* (1999, Prague, Czech Republic), vol. 1592 of *Lecture Notes in Computer Science*, Springer, pp. 223–238.

[95] Parno, B., Wendlandt, D., Shi, E., Perrig, A., Maggs, B., and Hu, Y.-C. Portcullis: Protecting Connection Setup from Denial-of-Capability Attacks. *ACM SIGCOMM Computer Communication Review 37*, 4 (2007), 289–300.

[96] Peng, T., Leckie, C., and Ramamohanarao, K. Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems. *ACM Computing Surveys 39*, 1 (2007), 3.

[97] Petraschek, M., Hoeher, T., Jung, O., Hlavacs, H., and Gansterer, W. Security and Usability Aspects of Man-in-the-Middle Attacks on ZRTP. *Journal of Universal Computer Science 14*, 5 (2008), 673–692.

[98] Petrovic, S., and Fúster-Sabater, A. An improved Cryptanalysis of the A5/2 Algorithm for Mobile Communications. In *Proceedings of the IASTED International Conference on Communication Systems and Networks* (2002, Malaga, Spain), pp. 437–444.

[99] Pohlig, S., and Hellman, M. An Improved Algorithm for Computing Logarithms over GF(p) and its Cryptographic Significance. *IEEE Transactions on Information Theory 24*, 1 (1978), 106–110.

[100] Pomerance, C. The Quadratic Sieve Factoring Algorithm. In *EUROCRYPT - Workshop on Advances in Cryptology* (1984, Paris, France), vol. 209 of *Lecture Notes in Computer Science*, Springer, pp. 169–182.

[101] Poupard, G., and Stern, J. Fair Encryption of RSA Keys. In *EUROCRYPT - Advances in Cryptology* (2000, Bruges, Belgium), vol. 1807 of *Lecture Notes in Computer Science*, Springer, pp. 172–189.

[102] Rivest, R. L., Shamir, A., and Adleman, L. A Method For Obtaining Digital Signatures And Public-Key Cryptosystems. *Communications Of ACM*, 2 (1978), 120–126.

[103] Savage, S., Wetherall, D., Karlin, A., and Anderson, T. Practical Network Support for IP Traceback. *ACM SIGCOMM Computer Communication*

Review 30, 4 (2000), 295–306.

[104] Schridde, C., and Freisleben, B. On the Validity of the Phi-Hiding Assumption in Asymmetric Cryptographic Protocols. In *ASIACRYPT - Advances in Cryptology* (2008, Melbourne, Australia), vol. 5350 of *Lecture Notes in Computer Science*, Springer, pp. 344–354.

[105] Schridde, C., and Freisleben, B. Partial Key Exposure Attacks on Secret Sharing Schemes. (submitted for publication).

[106] Schridde, C., Smith, M., Agel, B., and Freisleben, B. Secure Mobile Communication with Identity-based Cryptography and Cluster-aided Computations. *Journal of Supercomputing*, (to appear).

[107] Schridde, C., Smith, M., Dörnemann, T., Juhnke, E., and Freisleben, B. An Identity-Based Security Infrastructure for Cloud Environments. In *2010 IEEE International Conference on Wireless Communications Networking and Information Security* (2010, Peking, China), IEEE Press, p. (to appear).

[108] Schridde, C., Smith, M., and Freisleben, B. An Identity-Based Key Agreement Protocol for the Network Layer. In *SCN'08 - Proceedings of the 6th International Conference on Security and Cryptography for Networks* (2008, Amalfi, Italy), vol. 5229 of *Lecture Notes in Computer Science*, Springer, pp. 409–422.

[109] Schridde, C., Smith, M., and Freisleben, B. Non-Interactive Multi-Signatures with Multiple Independent Identity Key Generators. (submitted for publication).

[110] Schridde, C., Smith, M., and Freisleben, B. TrueIP: Prevention of IP Spoofing Attacks using Identity-based Cryptography. In *SIN'09 - Proceedings of the 2nd International Conference on Security of Information and Networks* (2009, Gazimagusa, North Cyprus), ACM Press, pp. 128–137.

[111] Schridde, C., Smith, M., Freisleben, B., and Kewitz, A. Verfahren und Vorrichtung für eine verschlüsselte digitale Sprachkommunikation, 2007. Patentanmeldung|DE|2007007257.

[112] Schridde, C., Smith, M., Freisleben, B., and Kewitz, A. Verfahren und Vorrichtung zur Erzeugung von kryptographischen Schlüsseln zur

Durchführung einer Schlüsseleinigung für eine sichere digitale Kommunikation, 2007. Patentanmeldung|DE|2007007248.

[113] Schridde, C., Smith, M., Freisleben, B., and Kewitz, A. Verfahren und Vorrichtung zur Erzeugung von kryptographischen Schlüsseln zur Durchführung einer Schlüsseleinigung für eine sichere digitale Kommunikation in einem IP-Netzwerk, 2007. Patentanmeldung|DE|2007007302.

[114] Schridde, C., Smith, M., Freisleben, B., and Kewitz, A. Verfahren und Vorrichtung zur kryptographischen Schlüsseleinigung für eine sichere digitale Kommunikation in IP-Netzwerken, 2007. Patentanmeldung|DE|2007007251.

[115] Shamir, A. How to Share a Secret. *Communications of the ACM 22*, 11 (1979), 612–613.

[116] Shamir, A. Identity-Based Cryptosystems and Signature Schemes. In *CRYPTO - Advances in Cryptology* (1984, Santa Barbara, California, USA), vol. 196 of *Lecture Notes in Computer Science*, Springer, pp. 47–53.

[117] Shim, K. Efficient ID-based Authenticated Key Agreement Protocol based on Weil Pairing. *Electronics Letters 39*, 8 (2003), 653–654.

[118] Shor, P. W. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In *FOCS - Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (1994, Santa Fe, New Mexico, USA), IEEE Press, pp. 124–134.

[119] Smart, N. The Discrete Logarithm Problem on Elliptic Curves of Trace One. *Journal of Cryptology 12*, 3 (1999), 193–196.

[120] Smetters, D. K., and Durfee, G. Domain-based Administration of Identity-Based Cryptosystems for Secure E-Mail and IPSEC. In *SSYM'03 - Proceedings of the 12th Conference on USENIX Security Symposium* (2003, San Antonio, Texas, US), USENIX Association, pp. 215–230.

[121] Smith, M., Schridde, C., and Freisleben, B. Identity-Based Cryptography for Securing Mobile Phone Calls. In *HWISE - Proceedings of the 5th IEEE International Workshop on Heterogeneous Wireless Sensor Networks* (2009, Bradford, UK), IEEE Press, pp. 23–29.

[122] Smith, M., Schridde, C., and Freisleben, B. Securing Mobile Phone Calls with Identity-Based Cryptography. In *ISA'09 - Proceedings of the 3rd International Conference on Information Security and Assurance* (2009, Seoul, Korea), Lecture Notes in Computer Science, Springer, pp. 124–134.

[123] Software, S. MIRACL - Multiprecision Integer and Rational Arithmetic C/C++ Library. http://www.shamus.ie/.

[124] Steiner, M., Tsudik, G., and Waidner, M. Diffie-Hellman Key Distribution Extended to Group Communication. In *CCS'96 - Proceedings of the 3rd ACM Conference on Computer and Communications Security* (1996, New Delhi, India), ACM Press, pp. 31–37.

[125] von zur Gathen, J., and Gerhard, J. *Modern Computer Algebra.* Cambridge University Press, New York, NY, USA, 1999.

[126] Waters, R. Efficient Identity-based Encryption without Random Oracles. In *EUROCRYPT - Advances in Cryptology* (2005, Aarhus, Denmark), vol. 3494 of *Lecture Notes in Computer Science*, Springer, pp. 114–127.

[127] Wiener, M. J. Cryptanalysis of Short RSA Secret Exponents. In *EUROCRYPT - Advances in Cryptology* (1990, Houthalen, Belgium), vol. 434 of *Lecture Notes in Computer Science*, Springer, pp. 372–392.

[128] Zimmermann, P., Johnston, A., and Callas, J. ZRTP: Media Path Key Agreement for Secure RTP, 2009. RFC Draft.

Lebenslauf

20.07.1978	geboren in Peine

Schulbildung

07/1985–06/1989	Grundschule, Freudenberg, Westfalen
07/1989–06/1998	Evangelisches Gymnasium, Siegen-Weidenau

Zivildienst

07/1998–08/1999	Friedenshort, Freudenberg, Westfalen

Studium/Promotion

10/1999–05/2005	Studium der Informatik an der Philipps-Universität in Marburg Diplomarbeit: *Lizenzierung und Trust-Management in einem service-orientierten ad-hoc Grid.* Betreuer: Prof. Dr. Bernd Freisleben
seit 06/2005	Wissenschaftlicher Mitarbeiter im Fachbereich Mathematik und Informatik an der Philipps-Universität Marburg Dissertation: *Secure Session Framework: An Identity-based Cryptographic Key Agreement and Signature Protocol.* Betreuer: Prof. Dr. Bernd Freisleben

Berufserfahrung

08/2003–10/2003	Praktikum bei Aventis Behring (Marburg), Abteilung *Preclinical Research.* Aufgabe: Funktionsoptimierung einer Reaktionskinetik
11/2003–12/2004	Aventis Behring (Marburg), Abteilung *Medical & Regulatory Systems.* Aufgabe: Arbeiten an der IT- und Web-Infrastruktur