

# Contributions to Provable Security and Efficient Cryptography

vom Fachbereich Informatik  
der Technischen Universität Darmstadt  
genehmigte

## Dissertation

zur Erreichung des akademischen Grades  
Doctor rerum naturalium (Dr. rer. nat.)

von

**Dipl.-Math. Katja Schmidt-Samoa**

aus Hagen, Westfalen

Referenten: Prof. Dr. Tsuyoshi Takagi (Future University–Hakodate)  
Prof. Dr. Johannes Buchmann (TU Darmstadt)

Datum der Einreichung: 9. November 2005

Datum der mündlichen Prüfung: 5. Januar 2006

Darmstadt, 2006  
Hochschulkenziffer: D17



*To my husband and my parents*



## Acknowledgements

First and foremost, I would like to thank Prof. Tsuyoshi Takagi for giving me the opportunity to work under his guidance, for enabling me to attend several international conferences, and for providing me access to the Japanese culture. I am pleased to thank Prof. Johannes Buchmann. He supervised this thesis as a second referee; and taking part in the activities of his group has always been interesting and enjoyable. Thank you for everything you taught me!

I spent an amazing time in Japan, and I am deeply grateful to all the people and organizations that made this experience possible. Prof. Kaoru Kurosawa has been a perfect host when I visited his research group at Ibaraki university, Hitachi, and I have benefitted very much from discussions with him. Thanks again here to Prof. Tsuyoshi Takagi, who encouraged me to apply for a stipend, and who welcomed me at Future University, Hakodate. The Japanese Society for the Promotion of Science (JSPS) liberally and generously funded this research internship.

Thanks also to Prof. Alexander May for renewing my contract. It was a pleasure to work for him; and we had numerous inspiring conversations. Sincere thanks for creating a pleasant working atmosphere to all my further colleagues—the present and the former ones—especially to Christina Lindenberg and Kai Wirt for taking the trouble to design our web-sites and administrate our network, and to Bodo Möller for fruitful discussions.

Many thanks to Marita Skrobic for her kindness and helpful support in non-scientific matters.

I owe more to my parents Luzie and—in memoriam—Hans-Peter Wedemeyer than to anyone else. I am grateful for their love, their endless patience, and for always encouraging me. Finally, I am deeply indebted my husband Tobias who always supported me in any possible way. Remarkably, his contributions have not only been non-academic, but extensive discussions about redundant number systems inspired some of the results in Chapter 6. It is hard to imagine how this work could have been finished without you, and therefore this thesis is dedicated to you.

## Abstract

This thesis deals with two main matters of modern public key cryptography: provable security and efficient implementation.

Indubitably, security is the most important property of any cryptographic scheme. Nevertheless, cryptographic algorithms have often been designed on a trial-and-error basis, *i. e.*, a system has been regarded as secure as long as it withstood cryptanalytic attacks. In contrast, the provable security approach provides rigorous mathematical proofs within well-defined models. Nowadays, provable security is a key requirement for many applications.

The main contribution of the first part of this thesis is the development and analysis of new provably secure trapdoor one-way permutations. (Trapdoor) one-way functions are the cardinal primitives in public key cryptography, as they are utilized as building blocks for numerous kinds of cryptographic protocols. For this reason, and because of the small number of promising candidates known today, the invention of new trapdoor one-way functions is of interest on its own. However, to prove the practical relevance of our proposal, we additionally invent several provably secure applications in the range of homomorphic encryption, fail-stop signature schemes, hybrid encryption, and trapdoor commitments.

In the second part of this work, we will turn our attention to the efficient implementation of public key algorithms. Besides security, efficiency is the main criterion when evaluating cryptographic schemes because inefficient cryptosystems are of little practical value. In widely-used hand-held devices with scarce resources, cryptosystems based on elliptic curve point groups are the first choice today. Consequently, it is an active area of research to enhance the efficiency of elliptic curve scalar multiplication, which is the most common operation in these cryptosystems. Our contribution here is located in the field of multiplication methods with low memory requirements. We will introduce an algorithm which is as efficient as the state-of-the-art solution, but which significantly reduces the consumption of working memory. Moreover, we will develop a highly flexible variant which can be adapted to the exact amount of available storage. Therefore, the algorithms presented here are especially useful in connection with limited-constraint devices such as smart-cards.

## Zusammenfassung

Diese Dissertation befasst sich mit zwei der wichtigsten Aspekte beim Entwurf von Public-Key Kryptosystemen: Beweisbare Sicherheit und Effizienz.

Es steht außer Frage, dass Sicherheit eine unverzichtbare Eigenschaft jeder kryptographischen Anwendung ist. Nichtsdestotrotz vertrauten die Entwickler kryptographischer Verfahren lange Zeit der Methode von Versuch-und-Irrtum, d.h. ein Verfahren galt so lange als sicher, bis erfolgreiche Kryptanalyse eine Schwäche offenbarte. Im Gegensatz dazu bedient sich der Ansatz beweisbarer Sicherheit strenger mathematischer Beweise in wohldefinierten Sicherheitsmodellen. Heutzutage gilt beweisbare Sicherheit in vielen Bereichen als eine Standardforderung an ein Kryptosystem.

Den Hauptbeitrag des ersten Teils dieser Arbeit bildet die Entwicklung und Analyse von neuen beweisbar sicheren Falltür-Einweg-Permutationen. (Falltür-)Einweg-Funktionen sind das wichtigste Primitiv in der Public-Key-Kryptographie, denn sie bilden den Hauptbaustein für zahlreiche kryptographische Anwendungen. Aus diesem Grunde, und nicht zuletzt weil nur äußerst wenige geeignete Kandidaten bekannt sind, ist die Entwicklung neuer (Falltür-)Einweg-Funktionen bereits für sich genommen von größtem Interesse. Um jedoch auch den praktischen Nutzen der vorgeschlagenen Konstruktionen unter Beweis zu stellen, werden beweisbar sichere Anwendungen aus den Bereichen homomorphe Verschlüsselung, Fail-Stop Signaturen, hybride Verschlüsselung und Falltür-Hinterlegungsverfahren vorgestellt und analysiert.

Der zweite Teil dieser Arbeit widmet sich der effizienten Implementierung von Algorithmen der Public-Key-Kryptographie. Neben der Sicherheit ist die Effizienz das Hauptbewertungsmerkmal kryptographischer Verfahren, denn ineffiziente Algorithmen sind unbrauchbar für praktische Anwendungen. In den heute weit verbreiteten mobilen Endgeräten mit beschränktem Speicherplatz sind Elliptische-Kurven-Kryptosysteme aufgrund ihrer moderaten Schlüssellängen die erste Wahl. Die häufigste Operation dieser Kryptosysteme ist die Skalarmultiplikation, daher ist die Entwicklung von Verfahren zu deren Beschleunigung ein vielbeachtetes Forschungsgebiet. Die vorliegende Arbeit beschäftigt sich speziell mit Multiplikations-Algorithmen, die geringe Anforderungen an den Speicherplatz stellen. Es wird ein neuer Algorithmus entworfen, der ebenso effizient ist wie die "state-of-the-art" Lösung, allerdings bei signifikant geringerem Speicherbedarf. Desweiteren wird eine Variante dieses Algorithmus entwickelt, die exakt an den zur Verfügung stehenden Speicherplatz angepasst werden kann, so dass keine wertvollen Ressourcen verschwendet werden müssen. Die vorgestellten Verfahren erweisen sich also als besonders nützlich für die Implementierung auf speicherbeschränkten Medien wie Smart-Cards.

## Erklärung<sup>1</sup>

Hiermit erkläre ich, dass ich die vorliegende Arbeit – abgesehen von den in ihr ausdrücklich erwähnten Hilfsmitteln – selbständig verfasst habe.

## Wissenschaftlicher Werdegang der Verfasserin in Kurzfassung<sup>2</sup>

Okt. 1992 – Sept. 2002	Studium der Mathematik mit Nebenfach Informatik an der Universität Kaiserslautern
23.9.2002	Diplomprüfung (Dipl.-Math.)
Nov. 2002 – März 2006	wissenschaftliche Mitarbeiterin am Fachgebiet Kryptographische Protokolle, Fachbereich Informatik, Technische Universität Darmstadt

---

<sup>1</sup>gemäß §9 Abs. 1 der Promotionsordnung der TU Darmstadt

<sup>2</sup>gemäß §20 Abs. 3 der Promotionsordnung der TU Darmstadt

# Contents

<b>1</b>	<b>Preface</b>	<b>1</b>
<b>I</b>	<b>Provable Security</b>	<b>5</b>
<b>2</b>	<b>Provable Security in Public Key Cryptography</b>	<b>7</b>
2.1	Introduction . . . . .	7
2.1.1	Reductionist Security . . . . .	8
2.1.2	The Random Oracle Model . . . . .	8
2.1.3	Limitations of Provable Security . . . . .	9
2.1.4	Notations . . . . .	10
2.2	Public Key Encryption . . . . .	11
2.2.1	Attack Models . . . . .	13
2.2.2	Attack Goals: One-wayness . . . . .	14
2.2.3	Attack Goals: Semantic Security and Indistinguishability . . . . .	15
2.3	Digital Signatures . . . . .	16
2.3.1	Attack Models . . . . .	17
2.3.2	Attack Goals . . . . .	18
2.3.3	The Hash-Then-Sign Paradigm . . . . .	19
<b>3</b>	<b>Improved and Explicit Security Reduction for RSA-OAEP and RSA-Paillier</b>	<b>21</b>
3.1	Introduction . . . . .	21
3.1.1	Related Work . . . . .	22
3.2	Security Reduction Algorithms of RSA-OAEP and RSA-Paillier . . . . .	23
3.2.1	RSA-OAEP . . . . .	23
3.2.2	RSA-Paillier . . . . .	26
3.3	The Proposed Reduction Algorithm . . . . .	26
3.3.1	Algorithm Lin_Cong . . . . .	30

3.3.2	Finding All Small Solutions . . . . .	32
3.3.3	Comparison with the Continuous Fraction Method . . . . .	33
3.4	Security Reduction Analysis using the Proposed Algorithm . . . . .	35
3.4.1	Application to RSA-OAEP . . . . .	35
3.4.2	Application to RSA-Paillier . . . . .	36
3.5	Conclusion . . . . .	38
<b>4</b>	<b>A New Rabin-type Trapdoor One-way Function and Its Applications</b>	<b>39</b>
4.1	Introduction . . . . .	39
4.1.1	Previous Work . . . . .	40
4.1.2	Our Contributions . . . . .	40
4.2	A Trapdoor One-way Permutation Equivalent to Factoring . . . . .	41
4.2.1	The Proposed Trapdoor One-way Function . . . . .	42
4.2.2	The Hardness of the $p^2q$ Factoring Problem . . . . .	45
4.2.3	Comparison . . . . .	45
4.3	Application to Homomorphic Encryption . . . . .	47
4.3.1	Basic Notions and Definitions for Homomorphic Encryption . . . . .	48
4.3.2	The Proposed Homomorphic Encryption Scheme . . . . .	49
4.3.3	Comparison . . . . .	52
4.4	Application to CCA Secure Hybrid Encryption . . . . .	53
4.4.1	Basic Notions and Definitions for Hybrid Encryption . . . . .	54
4.4.2	The Proposed Tag-KEM . . . . .	56
4.4.3	The Proposed Hybrid Encryption Scheme . . . . .	60
4.4.4	Comparison . . . . .	61
4.5	Application to Fail-stop Signature Schemes . . . . .	62
4.5.1	Basic Notions and Definitions for Fail-stop Signature Schemes . . . . .	64
4.5.2	The Proposed Fail-stop Signature Scheme . . . . .	69
4.5.3	Comparison . . . . .	72
4.6	Application to Trapdoor Commitment Schemes (Trapdoor Hash Families) . . . . .	72
4.6.1	Basic Notions and Definitions for Trapdoor Hash Families . . . . .	73
4.6.2	Recent Applications for Trapdoor Hash Families . . . . .	74
4.6.3	The Proposed Trapdoor Hash Families . . . . .	76
4.6.4	Comparison . . . . .	79
4.7	Conclusion . . . . .	80

---

<b>II</b>	<b>Efficient Implementation</b>	<b>81</b>
<b>5</b>	<b>Algorithms for Scalar Multiplication in Abelian Groups</b>	<b>83</b>
5.1	The General Scheme of Scalar Multiplication . . . . .	83
5.2	Known Solutions: The Generic Case . . . . .	85
5.3	Known Solutions: The Case Where Inversion Is Easy . . . . .	86
5.3.1	NAF And Windowed NAF . . . . .	87
5.3.2	$w$ NAF . . . . .	88
<b>6</b>	<b>MOF—A New Canonical Signed Binary Representation</b>	<b>91</b>
6.1	Introduction . . . . .	91
6.1.1	New Motivation for Memory-saving Scalar Multiplication Algorithms	92
6.1.2	Left-to-Right and Carry-free Generation Of Signed Representations	92
6.1.3	Related Works . . . . .	93
6.2	MOF: New Canonical Representation for Signed Binary Strings . . . . .	94
6.2.1	Converting Binary String to MOF . . . . .	95
6.3	Window Methods on MOF . . . . .	96
6.3.1	Right-to-Left Case: $w$ NAF . . . . .	96
6.3.2	Left-to-Right Case: $w$ MOF . . . . .	99
6.3.3	Comparison with Previous Methods . . . . .	101
6.4	Applications to Elliptic Curve Cryptography . . . . .	102
6.4.1	Elliptic Curve Scalar Multiplication . . . . .	102
6.4.2	Explicit Implementation . . . . .	103
6.5	Conclusion . . . . .	103
<b>7</b>	<b>Fractional MOF—Scalar Multiplication With Flexible Memory Consumption</b>	<b>105</b>
7.1	Introduction . . . . .	105
7.1.1	Elementary Markov Theory . . . . .	106
7.2	Fractional Window Methods . . . . .	107
7.2.1	Fractional $w$ NAF . . . . .	107
7.2.2	Fractional $w$ MOF . . . . .	109
7.3	Comparison . . . . .	113
7.4	Conclusion . . . . .	113
	<b>Bibliography</b>	<b>115</b>



# Chapter 1

## Preface

*"No one has yet discovered any warlike purpose to be served by the theory of numbers or relativity, and it seems very unlikely that anyone will do so for many years."*

G.H. Hardy, *The Mathematician's Apology*, 1940.

For centuries cryptography had almost exclusively been a business for governments and intelligence services, and cryptographic research had been a matter of utmost secrecy. In contrast, nowadays in the ubiquitous computing and networking area, cryptography is a matter of everybody's concern. In the course of these transitions, the objectives of cryptography (which literally translates to secret writing) became more multifarious. Instead of just enabling two parties to communicate secretly, in recent decades cryptographic methods have been developed to ensure multiple aspects of information security, as there are confidentiality, authenticity, data integrity, and anonymity. An important consequence of this changed meaning is the development of public key cryptography, a branch especially useful for open networking.

Since its foundation was provided by W. Diffie and M.E. Hellman [DH76], public key cryptography has fascinated mathematicians all over the world. One remarkable aspect of public key cryptography is the fact that it eventually establishes applications to the theory of numbers, one of the oldest and purest branches of mathematics. For millenia, number theory—considered as the queen of mathematics by C.F. Gauss—was exclusively conducted as an art for art's sake. In connection with public key cryptography, however, various applications emerged, not only immediate ones for elementary number theory but also indirect ones for algebraic number theory, *e. g.*, in the development of advanced factoring algorithms like the number field sieve [LL93]. Thus, we are faced with an impressive argument for the importance of fundamental research, though not everyone may be pleased with the actual state of affairs (see the quotation prefacing this chapter).

This thesis deals with two main topics of modern public key cryptography: provable security and efficiency. There is no doubt that security is the sine qua non of any cryptographic application. Nevertheless, cryptographic algorithms have often been designed on a trial-and-error basis, and security simply was defined as the absence of known attacks. As the history of cryptology shows, however, withstanding cryptanalytic attacks for a while provides no guarantee at all. In contrast, the provable security approach defines security in a framework adopted from computational complexity theory: a cryptographic scheme is said to be provably secure if any reasonable attack against its well-defined secu-

curity can be transferred into an algorithm solving a problem assumed to be hard, *e. g.*, the factorization of large integers. After various cryptanalytic achievements resulting in lots of broken schemes, nowadays provable security of cryptosystems is demanded as a crucial requirement even by standardization authorities [Sho04a, New04].

On the other hand, inefficient cryptosystems are of little practical value, regardless of the level of security they offer. To enhance the acceptance and popularity of public key cryptography, finding efficient solutions is a task of particular importance, especially with regard to the rapid dispersal of limited-constraint devices like smart-cards in the ubiquitous computing area.

Unfortunately, for a long time the issues of provable security and efficiency seemed to be incompatible. Indeed, the common property of all early provably secure cryptosystems was their high level of inefficiency. This went so far that in its beginning provable security has been regarded as a playground for theoreticians rather than a useful concept. As said before, the situation is different today. There are two reasons for this change in mind: the increased awareness of the necessity of reliable security arguments on the one hand, and the availability of efficient provably secure solutions on the other.

## Contributions Of This Thesis

In the first part of this thesis, we focus on the development and improvement of provably secure cryptographic schemes. After providing a quick account of provable security and establishing the most important notions in Chapter 2, we turn our attention to the security reductions of RSA-OAEP and RSA-Paillier in Chapter 3. In both proofs, the problem of finding small solutions of bivariate linear modular congruences arises and is solved by means of lattice reduction. We develop an alternative algorithm for this purpose, prove its correctness and efficiency, and finally apply it to both reduction proofs. In the case of RSA-OAEP, we are able to improve the reduction cost of the previously known solution, whilst for RSA-Paillier we obtain an explicit reduction algorithm and state the concrete reduction costs, where previous works provided asymptotic results only.

We consider Chapter 4 as the main contribution of this thesis. In this chapter, a new trapdoor one-way permutation based on the hardness of factoring integers of the shape  $n = p^2q$  is introduced and analyzed. We point out several similarities of our approach to Rabin-type modular squaring. There are, however, crucial differences, and we show that these discrepancies provide interesting applications to homomorphic encryption, hybrid encryption, fail-stop signature schemes and trapdoor commitments. In the following, we summarize our contributions to these different domains:

First, we prove that Paillier's famous homomorphic encryption scheme is one way under the  $p^2q$ -factorization assumption if we replace the modulus  $n = pq$  with  $n = p^2q$ . This is a notable improvement since Paillier's original scheme is proved to be one way under a non-standard assumption only. The flip-side of the coin, however, is that there is a simple IND-CCA1 attack mountable against our new variant. In case of Paillier's original scheme, it is unknown if IND-CCA1 security does hold or not. In environments where chosen-ciphertext attacks are not an issue the new scheme is clearly preferable.

Next, we propose a new hybrid encryption scheme following Abe *et al.*'s recently published Tag-KEM/DEM framework [AGK05]. Compared to the members of the well-known EPOC family, which are based on similar assumptions, the new scheme offers some advantages

in efficiency and parameter length.

Then we address the problem of constructing efficient factorization-based fail-stop signature schemes. All known solutions of provably secure fail-stop signature schemes follow a generic construction proposed by van Heijst, Pfitzmann and Pedersen in 1992 [vHPP92]. Utilizing the same framework, we develop a novel fail-stop signature scheme based on the  $p^2q$ -factorization assumption which is comparable in efficiency and parameter size to the state-of-the-art factorization-based solution [vHPP92]. In contrast to the latter, our new scheme is more elegant and simple. Moreover, we propose an improvement to the general construction, and based on this analysis it turns out that an adjusted version of our proposed scheme indeed outperforms the solution from van Heijst, Pfitzmann and Pedersen [vHPP92].

We conclude Chapter 4 with the development of two novel trapdoor commitment schemes. The first one, to the best of our knowledge, is the first trapdoor commitment that perfectly meets the requirements for Shamir-Taumann-type on-line/off-line signature schemes<sup>1</sup> [ST01]. The second one turns out to be the first factorization-based trapdoor commitment scheme that enables the construction of on-line/off-line chameleon signatures [KR00], thus improving a previous solution based on the  $\text{RSA}(n, n)$  problem [CGHGN01], *i. e.*, the hardness of inverting the RSA function where the public exponent equals the public modulus.

The second part of this thesis deals with efficient implementation of cryptographic algorithms in memory-constraint devices. Due to shorter key-lengths, elliptic curve cryptosystems (ECC) are the method of choice for these applications. The most important operation in ECC is scalar multiplication, *i. e.*, the multiplication of an elliptic curve point with an integer scalar. For this reason, efficient methods for performing this operation have been extensively studied by the cryptographic community in recent years. A review of the results most important within the scope of this thesis is given in Chapter 5. The basis for all common scalar multiplication methods is the so-called scalar recoding, that is, the re-writing of the binary scalar to a different base-2 representation with a reduced Hamming weight. As the binary representation is unique, this can only be achieved with digit sets different from  $\{0, 1\}$ .

In Chapter 6, we introduce a new signed binary representation of integers, *i. e.*, a base 2 representation utilizing the digit set  $\{0, \pm 1\}$ . In the case of ECC scalar multiplication, signed representations are preferable because the inversion of elliptic curve points can be computed virtually for free. Based on this representation—which we consider to be canonical for signed binary—we develop algorithms for signed representations with larger digit sets. Such representations are meaningful if precomputation is permitted. In contrast to all previous solutions, our new algorithm scans the scalar from the most to the least significant bit, *i. e.*, left-to-right. The benefit of this feature is that the common methods for evaluating the multiplication based on the preceding recoding also operate left-to-right, hence there is no longer any need to store the entire recoded scalar as an intermediate result. Moreover, it turns out that in terms of efficiency and precomputational effort, our proposed algorithm achieves the same properties as the best previously known (right-to-left) solution.

Finally, in Chapter 7, we apply Möller’s fractional window method [Möl02] to the representations introduced in Chapter 6. The aim of fractional windows is to maximize the effectiveness of memory consumption. More precisely, the fractional methods offer a

---

<sup>1</sup>Several working solutions exist in the literature, but our proposal combines the best known efficiency and the most powerful generic construction.

suitable multiplication algorithm exactly tailored for any amount of memory available, which has not been possible with previous solutions. We therefore obtain a highly flexible variant of the main algorithm from the preceding chapter. Moreover, using Markov theory, we are able to prove rigorously that the asymptotic non-zero density of the achieved representation is the same as Möller claimed in his original proposal for the right-to-left case. In a subsequent work independent from ours, Möller has also analyzed the left-to-right fractional window method [Möl04]. In particular, Möller has additionally proved minimality of Hamming weight among all representations utilizing a certain digit set, which is a further selling point for our proposed algorithm.

### **About This Thesis**

Except of the introducing Chapters 2 and 5, all work presented in this thesis has been published in conference proceedings or journals. Chapter 3 is taken from a paper published at Asiacrypt 2003 [KSST03]. The core part of this dissertation—Chapter 4—is based on three papers published at ICICS 2004 [SS04], STM 2005 [SS06] and MyCrypt 2005 [SST05]. Chapter 6 is extracted from a paper presented at Crypto 2004 [OSSST04a], and, finally, Chapter 7 is excerpted from an article to appear in IEEE Transactions on Computers [SSST06].

Some of the papers cited above are the outcome of collaborative research with my PhD supervisor Tsuyoshi Takagi and with my colleagues Kaoru Kurosawa, Katsuyuki Okeya, Olivier Semay, and Christian Spahn. To clarify my own contributions, theorems which have been provided by my co-authors are not proved in this thesis. Instead, a citation of the corresponding paper is given. This applies to Chapters 6 and 7.

**Part I**

**Provable Security**



## Chapter 2

# Provable Security in Public Key Cryptography

In this chapter, we review the most important notions related to provable security. We focus on those definitions that are important within the scope of this thesis. Fewer used terms are introduced when needed.

More than 50 years ago, when the rigorous theoretical treatment of security started with the pioneering work of Shannon [Sha48,Sha49], potential adversaries have been supposed to possess unlimited computational power. Withstanding these kind of attacks is denoted as *perfect secrecy* or (in modern terms) *information-theoretic security*. Perfectly secret encryption exists (*e.g.*, the well-known *Vernam's one-time pad*), it is, however, required that the communication parties exchange an appropriate amount of secret bits before the actual information transfer begins. Therefore, in modern cryptography (especially in the public key case where this burden has been overcome) the idea of an adversary with unbounded computational resources has been relaxed significantly. The current approach is related to practical infeasibility rather than absolute impossibility, in this sense a scheme informally is said to be secure if no polynomial-time attack with non-negligible success probability can be mounted against it. In contrast to information-theoretic security, we now talk about *computational security*. The bulk of this thesis considers computational security only, but in Section 4.5 we will see an example of a certain notion of information-theoretic security in the public key scenario.

The two main concepts of public key cryptography are encryption and digital signatures. Here, the security goals are privacy and authenticity, respectively. We start with some general remarks about the goals and limitations of provable security. Concepts related to encryption are reviewed in Section 2.2, whilst Section 2.3 is devoted to digital signatures.

## 2.1 Introduction

In the aftermath of the invention of public key cryptography by Diffie and Hellman in 1976 [DH76], design and evaluation of public key cryptosystems has been done merely in an ad-hoc manner based on trial-and-error. But very quickly, due to various cryptanalytic achievements resulting in lots of broken schemes, the cryptographic community understood

that this ad-hoc approach might not be good enough. Moreover, in the early days of public key cryptography, security considerations only dealt with the most basic attacks, *i. e.*, cryptanalytic research concentrated on inverting the schemes' underlying one-way functions. In the “real world”, however, cryptosystems normally flawed due to different reasons. The paradigm of provable security is an outcome of these insights. The goals of provable security are to define appropriate models of security on the one hand, and to develop cryptographic designs that can be proven to be secure within particular models on the other.

### 2.1.1 Reductionist Security

The roots of provable security go back to the early 1980s when the famous cryptographic researchers Goldwasser, Micali and Rivest adopted the framework from computational complexity theory for the purpose of rigorously defining the security of cryptographic schemes [GM82, GM84, GMR84]. In the so-called *reductionist security model*, a cryptosystem is called provably secure if there is a polynomial reduction proof from a well-established hard problem (such as the integer factorization problem) to an attack against the security of the cryptosystem. Informally, this means that if there is a polynomially bounded adversary breaking the scheme, then the problem assumed to be hard can also be solved in polynomial time. As this is a contradiction, no such adversary exists (provided the hardness assumption related to the problem is true). It remains to clarify the phrase *breaking the scheme* which depends on the particular security notion. In general, we distinguish different *attack goals* (*e. g.*, determining the plaintext from its ciphertext) and different *attack scenarios* (*e. g.*, availability of *oracles* that the adversary is allowed to query and which provide certain additional information like the decryption of ciphertexts of the adversary's choice). Needless to say, the strongest notion is defined as infeasibility of reaching the highest attack goal even in the most restricted attack scenario. In Sections 2.2 and 2.3, we will describe these matters in more detail. But first we will address a special attack scenario that has attracted much attention in modern practical-oriented cryptography.

### 2.1.2 The Random Oracle Model

Until the early 1990s, only rather inefficient solutions for cryptosystems with strong provable security were known. Consequently, provable security has been considered to be of theoretic interest only and has been ignored by standardization authorities at that time. Things changed 1993, when Bellare and Rogaway introduced the so called *random oracle model* (ROM) as a compromise between security requirements and practical efficiency considerations [BR93]. In the random oracle model, all parties—the legitimate ones as well as the adversary—have black-box access to functions which behave like truly random functions. Under this idealized assumption, it became possible to develop cryptosystems that are both efficient and provably secure. In concrete implementations, however, truly random functions are out of reach and the random oracles are replaced by concrete objects like cryptographic hash functions. Thus, it is obvious that even a rigorously analyzed security proof in the random oracle model does not guaranty security in the real world. A real world adversary may exploit some weaknesses of the hash functions used, thus a proof in the ROM can only exclude *generic* attacks against the scheme. Even worse, recently published results show separations between the random oracle scenario and the so-called

*standard model* where no random oracles are used, as there exist cryptosystems provably secure in the ROM that nevertheless are breakable when implemented with any concrete realization [CGH98, CGH04]. The interceders of the ROM, however, argue that those results do not affect the usability of the approach, because all found “weaknesses” only apply to artificial and pathological examples without practical relevance [Poi04, MK04]. In contemporary cryptographic research, the reliability of the random oracle paradigm is a topic of highly controversial discussion. Possibly most convincing is the viewpoint of D. Stinson, who argues that “[...] The random oracle model is just one of many possible attack models [...] It is not fundamentally “different” from many other attack models [...]” [Sti04]. The crucial point is that cryptographic security proofs *never* give evidence that a cryptographic protocol is secure in the “real world” (as we will see in the preceding paragraph 2.1.3), therefore the vast dispute about the role of the ROM seems to be disproportionate.

### 2.1.3 Limitations of Provable Security

Although mathematically elegant and appealing, the approach of proving security via reduction has some shortcomings. First of all, it is quite a young research area. Developing precise and adequate definitions of security related matters seems to be a delicate and challenging issue, and in certain cases it took several years until a widely accepted solution has been found. Consequently, definitional difficulties have been the source for multiple flawed schemes. Even the famous “gap” in the RSA-OAEP proof traces back to the use of an inadequate definition (see Section 3.2.1). A problem occurring frequently in this context is the clashing of theoretical attack scenarios with real world adversaries. A recent example is the security analysis of the well-known EPOC cryptosystems: EPOC-1 and EPOC-2 [FKM<sup>+</sup>00] have been proven to meet the strongest notion of security in the ROM. In the proofs, however, it is implicitly assumed that the adversary queries her decryption oracle on correctly generated ciphertexts only. In [JQY01], Joye *et al.* demonstrate that the EPOC cryptosystems can be completely broken if the adversary sends ciphertexts to her decryption oracle that she had constructed by encrypting ill-formed plaintexts. Their attack is similar to Bleichenbacher’s chosen ciphertext attack on PKCS #1 [Ble98]. In both cases, the information if cleverly constructed ciphertexts are valid or not can be exploited to obtain secret data via binary search. In contrast to PKCS #1, there is a simple fix available for the EPOC cryptosystems.

Another notable point is that reductionist security proofs are of asymptotic nature only. Usually, a so-called security parameter  $k$  is established and the proved result is of the following kind: if the appropriate system parameters (*e. g.*, key-lengths) equal  $k$ , and if there is an adversary  $\mathcal{A}$  against the scheme with running time polynomial in  $k$ , then the underlying problem assumed to be hard can also be solved in time polynomial in  $k$ . The proof usually is done by presenting a problem solver which utilizes the presumed adversary as a subroutine, such that the former runs in polynomial time of the latter. A positive interpretation of such a result is that the analyzed scheme is reliable if we just choose the security parameter large enough. The devil’s advocate, however, may argue that asymptotic security provides no evidence for any concrete instantiation, where naturally the security parameter is fixed. This problem is the more serious the larger the gap between breaking the original scheme and solving the underlying problem actually is, *i. e.*, the worse the reduction is. Here, the optimal case is referred to as *tight security reduction*,

defined as a reduction that carries over one instance into the other without any reasonable loss of efficiency or success probability. To deal with the problem of asymptotic security, recently a new line of cryptographic research came up, the so-called *concrete security* approach (for details see the survey paper [Bel97]). A concrete security reduction is intended to provide *quantitative* results instead of just *qualitative* ones. That is, the success probability of the attack against the trusted assumption is exactly determined as a function of the adversary  $\mathcal{A}$ 's success probability and of the amount of additional information  $\mathcal{A}$  needs for breaking the scheme (*e.g.*, the number of decryption queries in the accordant model). In an analogue way, the running times are related to each other. The objective of this attempt apparently is to give practitioners concrete numbers at hand, in order to facilitate the optimal choice of the security parameter. However, it is often overlooked that recommendations of key sizes are always heuristics, regardless of how exactly the reduction is analyzed. This is due to the fact that the run time analysis of algorithms for solving the most important candidates for fundamental hard problems (*e.g.*, integer factorization or discrete logarithm) are themselves of asymptotic nature only. Thus, it is out of reach to determine exactly how many basic operations<sup>1</sup> are required to factor say a 2048 bit RSA modulus. In addition, when translating the concrete reduction into a parameter size recommendation, the amount of additional information the potential adversary asks for has to be specified. For example, it is common to assume that a chosen-ciphertext adversary in the random oracle model makes at most  $2^{60}$  oracle- and  $2^{30}$  decryption-queries, but these values are just arbitrary estimations. With these facts in mind, some scepticism against conclusions like “the key-length should be chosen not smaller than 1139 bits” should be advisable. Concrete security may have its benefits compared with asymptotics only, but it is not a definite solution of all the problems, and it should be avoided to sham accuracy by providing too explicit recommendations. In Section 4.4, we will give an example for a concrete security reduction.

Last but not least, it has to be mentioned that security proofs do not rule out attacks on the implementation level. The proofs only deal with *black-box reductions*, *i. e.*, the schemes are treated as algorithms specified by input-output behavior only. In concrete implementations, however, additional information may be leaked like power consumption, timings of operations, electromagnetic emanations, etc. Attacks based on this kind of information are called *side channel attacks*. Side channel attacks and similar *fault attacks*—where the adversary tries to gain informations about secret parameters by actively inducing errors into the cryptographic algorithms—are a topic of active research, but are not dealt with in this thesis (see [QK02] for a fairly actual survey; the interested reader may also check out the web-site of the Side Channel Cryptanalysis Lounge [SCA] for further information).

## 2.1.4 Notations

Throughout this thesis, we use the following notations:

The magnitude of a finite set  $S$  is denoted with  $\#S$ .

Let  $n$  be a positive integer. We write  $\mathbb{Z}_n$  for the ring of residue classes modulo  $n$ , and  $\mathbb{Z}_n^\times$  for its multiplicative group, *i. e.*, the set of invertible elements modulo  $n$ . Unless stated otherwise, we use  $\{0, 1, \dots, n-1\}$  as representatives for the residue classes modulo  $n$ .

For  $x \in \mathbb{Z}_n^\times$ ,  $\text{ord}_n(x)$  denotes the multiplicative order of  $x$  modulo  $n$ , *i. e.*, the smallest

<sup>1</sup>In the literature about concrete security, a basic operation usually refers to a the equivalent of a 3-DES encryption.

positive integer  $k$  with  $x^k = 1 \pmod n$ .

Furthermore,  $\varphi : \mathbb{N} \rightarrow \mathbb{N}$  and  $\lambda : \mathbb{N} \rightarrow \mathbb{N}$  are Euler's totient function and Carmichael's function, respectively.

The  $L$ -function is defined as  $L_n(x) = (x-1)/n$  for  $n \in \mathbb{N}$  and  $x \in \{x \in \mathbb{Z} \mid n \text{ divides } x-1\}$ . For any homomorphism  $h$ , we denote its kernel and its image with  $\ker(h)$  and  $\text{im}(h)$ , respectively.

Unless indicated otherwise, all algorithms are randomized, but we do not mention the random coins as an extra input. If  $A$  is a probabilistic algorithm, then  $A(y_1, \dots, y_n)$  refers to the probability space which to the string  $x$  assigns the probability that  $A$  on input  $y_1, \dots, y_n$  outputs  $x$ . For any probability space  $S$ , the phrase  $x \leftarrow S$  denotes the experiment that  $x$  is selected at random according to  $S$ . In particular, if  $S$  is a finite set, then  $x \leftarrow S$  is the operation of picking  $x$  uniformly at random from  $S$ . If  $S_1, S_2, \dots, S_n$  are probability spaces and  $p$  is an  $n$ -ary predicate, then  $\Pr[x_1 \leftarrow S_1; \dots; x_n \leftarrow S_n : p(x_1, \dots, x_n)]$  denotes the probability that  $p(x_1, \dots, x_n)$  is true after the experiments  $x_i \leftarrow S_i$  have been executed in ascending order.

PPA is an abbreviation of probabilistic polynomial time algorithm.

$\text{PRIMES}$  is the set of all prime numbers, and  $\text{PRIMES}(l)$  denotes the set of all prime numbers with bit-length  $l$ .

The logarithm  $\log$  always refers to base 2.

We write  $|x|_2$  for the bit-length of the integer  $x$ . In addition, if  $x$  is a natural number, we write  $[x]^l$  for the  $l$  most significant bits and  $[x]_l$  for the  $l$  least significant bits of the binary representation of  $x$ , respectively. These values may be identified with the corresponding integers.

## 2.2 Public Key Encryption

In the following, we review security notions related to public key encryption. First of all, we formally define a public key encryption scheme:

**DEFINITION 2.1 (PUBLIC KEY ENCRYPTION SCHEME)** *A public key encryption scheme is a triple  $(\text{KeyGen}, \mathcal{E}, \mathcal{D})$  of polynomial time algorithms such that the following holds:*

1.  $\text{KeyGen}$  is a probabilistic key generation algorithm which on input  $1^k$  (the security parameter) produces a pair  $(\text{pk}, \text{sk})$  of public and secret key. For notational simplicity, we assume that the public key  $\text{pk}$  implicitly includes a description of the finite space of messages  $\mathcal{M}$  and the finite space of ciphertexts  $\mathcal{C}$ .
2.  $\mathcal{E}$  is a probabilistic encryption algorithm which on input  $m \in \mathcal{M}$  and  $\text{pk}$  produces a ciphertext  $c \in \mathcal{C}$ . For convenience, we also write  $\mathcal{E}_{\text{pk}}(m)$  instead of  $\mathcal{E}(m, \text{pk})$ .
3.  $\mathcal{D}$  is a deterministic decryption algorithm which on input  $c \in \mathcal{C}$  and  $\text{sk}$  outputs a message  $m \in \mathcal{M}$  or a failure symbol  $\perp$  indicating that the input ciphertext is invalid. For convenience, we also write  $\mathcal{D}_{\text{sk}}(c)$  instead of  $\mathcal{D}(c, \text{sk})$ .
4. If  $(\text{pk}, \text{sk})$  is a possible outcome of  $\text{KeyGen}$ , then we have  $\mathcal{D}_{\text{sk}}(\mathcal{E}_{\text{pk}}(m)) = m$  for all  $m \in \mathcal{M}$ .

**REMARK 2.1** In the definition above, the random coins used for encryption are not mentioned as an additional input. This notation simplifies most preceding definitions. When

describing explicit probabilistic encryption schemes, however, it is convenient to specify the randomness, too. In this case, we separate the message from the randomness with a semicolon:  $\mathcal{E}_{\text{pk}}(m; r)$ .

**EXAMPLE 2.1 (THE RSA ENCRYPTION SCHEME)** The most famous public key encryption system is the RSA cryptosystem, named after the mathematicians Rivest, Shamir and Adleman who invented it in 1977 [RSA78]. It can be described as follows:

**KeyGen** : On the input  $1^k$  (security parameter) choose  $p, q \in \text{PRIMES}(k)$  with  $p \neq q$  and define  $n = pq$ . Furthermore, select a small  $e$  relatively prime to  $\varphi(n) = (p-1)(q-1)$  and define  $d = e^{-1} \bmod \varphi(n)$ . The public key equals  $(n, e)$ , whilst  $d$  is the secret key. Moreover, we have  $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$ .

$\mathcal{E}$  : For  $m \in \mathcal{M}$ , the ciphertext  $c$  is defined as  $c = m^e \bmod n$ .

$\mathcal{D}$  : A ciphertext  $c \in \mathcal{C}$  is decrypted as  $m = c^d \bmod n$ .

The decryption procedure is correct because

$$(m^e)^d = m^{1+l\varphi(n)} = m \bmod n$$

holds for a suitable  $l \in \mathbb{Z}$ .

Note that RSA encryption is deterministic, although we allow the encryption algorithm to be probabilistic, *i. e.*, encrypting the same message several times may lead to pairwise different ciphertexts. One obvious advantage of probabilistic encryption appears if we assume that the message space is quite small (*e. g.*, four digit numbers for credit card PINs): If the encryption algorithm were deterministic, recovering the original message from a given cipher could be easily done by encrypting all possible plaintexts and comparing the results with the cipher in question. Moreover, if encryption is deterministic, sending the same message twice is easily detectable; this is undesirable in most applications. Another important but less obvious benefit of probabilistic encryption is postponed to Section 2.2.3.

We give a second example to picture probabilistic encryption:

**EXAMPLE 2.2 (PAILLIER'S ENCRYPTION SCHEME)** Although probabilistic encryption is known since 1984 (see Section 2.2.3), we present a rather young example which is important in subsequent chapters of this thesis. The following scheme has been proposed by Pascal Paillier in 1999 [Pai99a].

**KeyGen** : On the input  $1^k$  (security parameter) choose  $p, q \in \text{PRIMES}(k)$  with  $p \neq q$  and define  $n = pq$ . Furthermore, choose  $g \in \mathbb{Z}_{n^2}^\times$  such that  $n \mid \text{ord}_{n^2}(g)$  and define  $\lambda = \text{lcm}(p-1, q-1)$ . The public key equals  $(n, g)$ , whilst  $\lambda$  is the secret key<sup>2</sup>. Moreover, set  $\mathcal{M} = \mathbb{Z}_n, \mathcal{C} = \mathbb{Z}_{n^2}$ .

$\mathcal{E}$  : Choose a random value  $r \in \mathbb{Z}_n^\times$ . For  $m \in \mathcal{M}$ , the ciphertext  $c$  is defined as  $c = g^{m_r^n} \bmod n^2$ .

$\mathcal{D}$  : A ciphertext  $c \in \mathcal{C}$  is decrypted as  $m = \frac{L_n(c^\lambda \bmod n^2)}{L_n(g^\lambda \bmod n^2)} \bmod n$ , where the  $L$ -function here is defined as  $L_n(x) = (x-1)/n$

---

<sup>2</sup>Note that  $\lambda$  equals the evaluation of Charnichael's function on  $n$ .

In [Pai99a], it is shown that the decryption procedure is correct, and especially that the values of  $c^\lambda \bmod n^2 - 1$  and  $g^\lambda \bmod n^2 - 1$  are indeed divisible by  $n$ . It is notable that Paillier's scheme possesses *homomorphic properties*, *i. e.*, we have  $(g_1^m r_1^n)(g_2^m r_2^n) = g^{m_1+m_2}(r_1 r_2)^n \bmod n^2$ . For efficiency, it is useful to choose  $g = n + 1$  (note that we have  $\text{ord}_{n^2}(n+1) = n$ , thus  $g = n+1$  is admissible). This choice speeds up encryption because of  $(n+1)^m = 1 + mn \bmod n^2$ . Further important variants include RSA-Paillier [CGHGN01] and Rabin-Paillier [GMMV03].

Naturally, we want an encryption scheme to assure a certain kind of privacy. In the rest of this section, we develop the definitions to capture different levels of this issue. As we only deal with computational security, we have to come up with a notion of practical infeasibility, *i. e.*, we want to express that no polynomial-time attack can be mounted with non-negligible success probability. Thus, we need the definition of a negligible function:

**DEFINITION 2.2 (NEGLIGIBLE FUNCTION)** *A function  $f : \mathbb{N} \rightarrow \mathbb{R}^{\geq 0}$  is called negligible if  $f$  decreases faster than the reciprocal of any polynomial, *i. e.*, for every constant  $c > 0$  there exists an integer  $k_c$  such that  $f(k) < k^{-c}$  for all  $k \geq k_c$ .*

### 2.2.1 Attack Models

As stated before, we evaluate the security of an encryption scheme by distinguishing different attack goals and different attack scenarios. We start with examining the latter. Here, the most crucial distinction is made between *active* and *passive* attacks. A passive adversary is only allowed to encrypt messages of her choice, therefore this attack scenario is known as *chosen-plaintext attack* (CPA). Note that in the public key scenario, everyone is in the position to encrypt arbitrary messages, thus this scenario is indeed the weakest one.

In contrast, active attackers are assumed to have the ability of decrypting ciphertexts of their choice, therefore this model is called *chosen-ciphertext attack* (CCA). If not specified otherwise, the adversary is allowed to put decryption queries in an *adaptive* manner, *i. e.*, the actual query may depend on all of the preceding ones.

In any case, the ability of encrypting plaintexts—resp. decrypting ciphertexts—is modeled as the availability of corresponding *oracles* for the adversary. This is meant by *black box* access to the encryption scheme.

The general framework is similar for most security definitions: The attack is formalized as a game played by the adversary. She is given the public data and interacts with her oracles as specified in the concrete attack scenario. At a certain point, the adversary indicates that the first phase is finished and then, she is given an appropriate *challenge* which she tries to solve, again interacting with the available oracles. The scheme is said to be secure if no polynomial-time adversary wins the attack game with a non-negligible success probability.

## 2.2.2 Attack Goals: One-wayness

One of the weakest demands on encryption schemes is that it should be infeasible to recover the entire message from its corresponding ciphertext without knowledge of the secret key<sup>3</sup>. This requirement is denoted *one-wayness* (OW).

To define one-wayness formally we use the framework sketched above. Here the challenge simply is a randomly generated ciphertext. Thus, we have the following definition:

**DEFINITION 2.3 (OW-CPA, OW-CCA)** *Let  $PKE = (\text{KeyGen}, \mathcal{E}, \mathcal{D})$  be a public key encryption scheme and let  $\mathcal{A}$  be an adversary against the one-wayness of  $PKE$  playing the following game:*

**GAME.OW:**

Step 1.  $(pk, sk) \leftarrow \text{KeyGen}(1^k)$

Step 2.  $m \leftarrow \mathcal{M}$

Step 3.  $c \leftarrow \mathcal{E}_{pk}(m)$

Step 4.  $\tilde{m} \leftarrow \mathcal{A}^{\mathcal{O}}(pk, c)$

*The superscript  $\mathcal{O}$  indicates the oracle(s) the adversary is allowed to query. In case of chosen-plaintext attack,  $\mathcal{O}$  stands for an encryption oracle only, whereas in case of chosen-ciphertext attack  $\mathcal{O}$  includes encryption oracle and decryption oracle as well. In the ROM,  $\mathcal{O}$  additionally includes access to the random oracles. However,  $\mathcal{A}$  is not allowed to query her decryption oracle on the challenge ciphertext. We define the advantage of the adversary  $\mathcal{A}$  as  $\epsilon_{\mathcal{A}} = \Pr[\tilde{m} = m]$ . Moreover,  $\epsilon$  is defined as  $\max_{\mathcal{A}}(\epsilon_{\mathcal{A}})$ , where the maximum is taken over all adversaries modeled as probabilistic polynomial time Turing machines.  $PKE$  is said to be one-way against chosen-plaintext attacks (OW-CPA) or one-way against chosen-ciphertext attacks (OW-CCA), respectively, if  $\epsilon$  is negligible in the security parameter  $k$ .*

Chosen-ciphertext one-wayness is a strictly stronger notion than the chosen-plaintext one, as the RSA cryptosystem demonstrates: RSA is assumed to be OW-CPA (this claim is well-known as *RSA assumption*), but due to its homomorphic properties it is vulnerable against chosen-ciphertext attacks: given the challenge  $c \in \mathcal{C}$ , the attacker chooses a random value  $r \in \mathbb{Z}_n^{\times}$  and feeds her decryption oracle with  $r^e c \bmod n$ . From the received value  $(r^e c)^d = rm \bmod n$  the message  $m$  can be easily obtained by multiplication with  $r^{-1} \bmod n$ .

One-wayness usually is not strong enough. First, one-wayness security does not prevent attackers from reconstructing messages that are taken from a small subset of  $\mathcal{M}$  (As it is mentioned in [GB01], the amount of words in  $\{A, B, \dots, Z\}^k$  that form correct sentences in English language is negligible if  $k$  is large enough. Thus, even a secure one-way encryption scheme may fail if only highly structured messages are encrypted.) Second, one-wayness security is only related to the *entire* message, it does not rule out the possibility of obtaining some specific partial information about the encrypted message. In the RSA case, for

---

<sup>3</sup>We do not state *the* weakest demand, because the notion of withstanding total breaks is even weaker, where a total break is defined as computing the secret key from the public data. We do not discuss total breaks in this thesis.

instance, the Jacobi Symbols of the plaintext and the corresponding ciphertext are equal (since  $e$  is odd), thus this information is not protected when using RSA encryption.

For these reasons we need a stronger notion: semantic security will do.

### 2.2.3 Attack Goals: Semantic Security and Indistinguishability

The notion of *semantic security* (SS) has been invented in a seminal paper by Goldwasser and Micali [GM84]. Roughly speaking, an encryption scheme is semantically secure if no meaningful information about the encrypted message can be computed from its ciphertext. This notion is the analogue of Shannon's perfect secrecy in the computational model. Goldwasser and Micali noticed that no deterministic encryption scheme can be semantically secure, and consequently they introduced the concept of probabilistic encryption. Defining semantic security formally, however, is somewhat involved, and for the sake of proving cryptosystems to be meet this strong requirement, the notion of *indistinguishability* (IND) turned out to be more convenient. The latter means that given a ciphertext and two possible plaintexts, it is infeasible to decide which one has been encrypted. Indistinguishability has also been introduced by Goldwasser and Micali in [GM84], where they proved equivalence between SS and IND for the chosen-plaintext attack scenario. Interestingly, the analogue equivalence for the chosen-ciphertext model has been assumed to be cryptographic folklore for years, but was formally proved only very recently [WSI03, Gol03]. As now equivalence is shown for all attack scenarios, we only deal with indistinguishability in this thesis, and sometimes we may use the phrase semantic security as a synonym for both notions.

To define indistinguishability formally, again we use the game-oriented framework. In contrast to one-wayness, the IND game is divided in two phases: in the *find stage* the adversary given the public key chooses two messages and in the *guess stage*, she has to decide which of these has been encrypted to form the challenge. Thus, we have the following definition:

**DEFINITION 2.4 (IND-CPA, IND-CCA)** *Let  $PKE = (\text{KeyGen}, \mathcal{E}, \mathcal{D})$  be a public key encryption scheme and let  $\mathcal{A}$  be an adversary against the indistinguishability of PKE playing the following game:*

**GAME.IND:**

Step 1.  $(pk, sk) \leftarrow \text{KeyGen}(1^k)$

Step 2.  $(m_0, m_1, st) \leftarrow \mathcal{A}^{\mathcal{O}}(pk)$

Step 3.  $b \leftarrow \{0, 1\}$

Step 4.  $c \leftarrow \mathcal{E}_{pk}(m_b)$

Step 5.  $\tilde{b} \leftarrow \mathcal{A}^{\mathcal{O}}(c, st)$

*The superscript  $\mathcal{O}$  indicates the oracle(s) the adversary is allowed to query. In case of chosen-plaintext attacks,  $\mathcal{O}$  stands for an encryption oracle only, whereas in case of chosen-ciphertext attacks,  $\mathcal{O}$  includes encryption oracle and decryption oracle as well. In the ROM,  $\mathcal{O}$  additionally includes access to the random oracles. In Step 5, however,  $\mathcal{A}$  is not allowed*

to query her decryption oracle on the challenge ciphertext. The value  $st$  is an internal state information. We define the advantage of the adversary  $\mathcal{A}$  as  $\epsilon_{\mathcal{A}} = \left| \Pr[\tilde{b} = b] - \frac{1}{2} \right|$ . Moreover, we define  $\epsilon$  as  $\max_{\mathcal{A}}(\epsilon_{\mathcal{A}})$ , where the maximum is taken over all adversaries modeled as probabilistic polynomial time Turing machines. PKE is said to be indistinguishable under chosen-plaintext attacks (IND-CPA) or indistinguishable under chosen-ciphertext attacks (IND-CCA), respectively, if  $\epsilon$  is negligible in the security parameter  $k$ .

Again, chosen-ciphertext attacks turn out to be strictly stronger than chosen-plaintext ones. We demonstrate this fact with Paillier’s cryptosystem. On the one hand, Paillier proved his scheme to be IND-CPA under the so-called *Decisional Composite Residuosity Assumption*, on the other hand a chosen-ciphertext attacker can easily win GAME.IND: From  $g^{m+1}r^n = g(g^m r^n) \bmod n^2$  we deduce that the ciphertext of  $m + 1$  is obtained by multiplying the ciphertext of  $m$  with the public value  $g$ . This shows that under a chosen-ciphertext attack Paillier’s cryptosystem is not even one-way<sup>4</sup>.

REMARK 2.2 There is as well a refinement as a relaxation of the definition of IND-CCA known. On the one hand, a further subdivision can be made by restricting the adversary’s access to the decryption oracle. Namely, a strictly weaker attack scenario is obtained if the adversary is only allowed to query this oracle during the find-stage, *i. e.*, before she is given the challenge ciphertext. This model is often called *lunchtime attack*, and sometimes it is referred to as IND-CCA1 in contrast to the above described general case, which for clarification is denoted IND-CCA2. In this thesis, when speaking of IND-CCA resp. chosen-ciphertext attacks, we always think of the stronger notion. The same treatment also applies to the notion of one-wayness. Here again, chosen-ciphertext attacks of lunchtime-type are strictly weaker than unrestricted ones.

A relaxation of IND-CCA security, on the other hand, has been recently proposed by An *et al.* [ADR02]. An *et al.* complain about a definitional inadequacy in the above given definition, namely an IND-CCA secure encryption scheme becomes formally “insecure” if each ciphertext is padded with a useless random bit. Indeed, in this case the adversary can switch the last bit of her challenge ciphertext and feed her decryption oracle with the altered cipher. For the sake of ruling out such unintuitive counterexamples, An *et al.* define a generalized notion of IND-CCA which they call gIND-CCA. As the definitional inadequacy does not affect the results presented in this thesis, we stick to the standard notion nevertheless.

## 2.3 Digital Signatures

Digital signatures are intended to replace handwritten signatures in the electronic world. The security goal here is authenticity, *e. g.*, the proof of authorship of messages. Besides obvious applications in electronic commerce, digital signatures are important building blocks for various kinds of cryptographic protocols, and traditional public key infrastructures rely on digital signatures for certifying public keys. We define a digital signature scheme as follows:

<sup>4</sup>The above described weakness is related to the notion of *non-malleability* (NM) [DDN91]. Roughly speaking, a cryptosystem is non-malleable if given a ciphertext  $c$  it is infeasible to generate a ciphertext  $c' \neq c$  such that the two underlying plaintexts are “meaningfully related”. It can be shown that IND-CCA and NM-CCA are equivalent notions, whilst NM-CPA is strictly stronger than IND-CPA [BDPR98].

**DEFINITION 2.5 (DIGITAL SIGNATURE SCHEME)** *A digital signature scheme is a triple  $(\text{KeyGen}, \mathcal{S}, \mathcal{V})$  of polynomial time algorithms such that the following holds:*

1. *KeyGen is a probabilistic key generation algorithm which on input  $1^k$  (the security parameter) produces a pair  $(\text{sk}, \text{vk})$  of secret signing and public verification key. For notational simplicity, we assume that the verification key  $\text{vk}$  implicitly includes a description of the finite spaces of messages  $\mathcal{M}$  and signatures  $\text{Sig}$ .*
2.  *$\mathcal{S}$  is a probabilistic signature algorithm which on input  $m \in \mathcal{M}$  and  $\text{pk}$  produces a signature  $\sigma \in \text{Sig}$ . For convenience, we also write  $\mathcal{S}_{\text{sk}}(m)$  instead of  $\mathcal{S}(m, \text{vk})$ .*
3.  *$\mathcal{V}$  is a deterministic verification algorithm which on input  $\sigma \in \text{Sig}, m \in \mathcal{M}$  and  $\text{vk}$  outputs “accept” or “reject” indicating if  $\sigma$  is a valid signature on  $m$ . For convenience, we also write  $\mathcal{V}_{\text{vk}}(\sigma, m)$  instead of  $\mathcal{V}(\sigma, m, \text{vk})$ .*
4. *If  $(\text{sk}, \text{vk})$  is a possible outcome of KeyGen, then for all  $m \in \mathcal{M}$  we have  $\mathcal{V}_{\text{vk}}(\sigma, m) = \text{“accept”}$  iff  $\sigma$  has been produced via  $\sigma \leftarrow \mathcal{S}_{\text{sk}}(m)$ .*

**EXAMPLE 2.3 (RSA SIGNATURES)** It has already been observed by Diffie and Hellman in their seminal paper [DH76] that there is a dual signature scheme for each public key encryption scheme build from a trapdoor one-way permutation like the RSA function (see Chapter 4 for more on trapdoor one-way permutations). In the RSA case this leads to the following signature scheme:

**KeyGen :** The key generation is the same as for the RSA encryption scheme (Example 2.1).  
The verification key equals  $(n, e)$ , whilst  $d$  is the signing key. Moreover, we have  $\mathcal{M} = \text{Sig} = \mathbb{Z}_n$ .

**$\mathcal{S}$  :** For  $m \in \mathcal{M}$ , the signature  $\sigma$  is defined as  $\sigma = m^d \bmod n$

**$\mathcal{V}$  :**  $\sigma$  is a valid signature of  $m$  iff  $m = \sigma^e \bmod n$  is fulfilled.

Again, the above example is only of deterministic nature, although the definition supports probabilistic signing. The standardized RSA-PSS [BR96]—one of the most popular digital signature scheme today—is a probabilistic variant of the above described “plain” RSA signature scheme<sup>5</sup>. Further well-known probabilistic signature schemes are the classical El-Gamal [Gam84] and its variants.

In the following, we introduce different attack models and possible attack goals in order to define the security of a signature scheme formally. We do not provide an exhaustive overview, however (see [GMR88] instead).

### 2.3.1 Attack Models

As in the public key encryption case, we define different security levels by providing the attacker with some sort of additional information. In any case, it is assumed that the attacker knows the public data, *i. e.*, the verification key and the system parameters. If no further information is given to the adversary, the corresponding attack is denoted *no message attack*, *a. k. a. key-only attack*.

<sup>5</sup>Indeed, PSS is an abbreviation of probabilistic signature scheme.

On the contrary, in a *message attack*, the adversary is allowed to examine some valid message/signature pairs. We distinguish the following scenarios in increasing order with respect to severity:

*Known-message attack:* The adversary is given a set of valid message/signature pairs randomly chosen by the challenger.

*Generic chosen message attack:* The adversary is allowed to choose a list of messages  $m_1, \dots, m_k$  and to query for the corresponding signatures *before* she knows the verification key. In addition, the queries have to be non-adaptively, *i. e.*, the adversary has to query the entire set at once.

*Adaptive chosen-message attack:* This is the most powerful attack. The adversary can use the legitimate signer as a signing oracle, *i. e.*, she may ask adaptive signature queries after seeing the verification key.

### 2.3.2 Attack Goals

A signature scheme is broken if it is feasible for an attacker to forge a signature, *i. e.*, to come up with a signature/message pair that passes the verification test. We distinguish the following cases in decreasing order with respect to dangerousness:

*Total break:* The adversary succeeds in computing the secret key.

*Selective forgery:* The adversary creates a valid signature for a message of her choice.

*Existential forgery:* The adversary creates any valid message/signature pair, without having control over the message.

Needless to say, in case of a message attack the forged signature must be different from any signature the adversary has been given earlier. As the signature algorithm may be probabilistic, this requirement is somewhat vague. Is a new signature  $\tilde{\sigma}$  on a message  $m$  such that the adversary has been given  $m, \sigma$  during the attack regarded as a valid forgery or not? As this kind of forgery usually does no harm, the answer is “No” in general. If nevertheless the case where the answer is “Yes” should be captured, the phrase “strong” is appended for clarification (*e. g.*, sUF-CMA denotes strong existentially unforgeability under chosen-message attacks).

For the sake of easy readability, we only define the most important and strongest notion formally (obtained by combining the strongest attack with the weakest goal). All other combinations of attack model/attack goal can be defined in an analogue way.

**DEFINITION 2.6 (UF-CMA)** *Let  $DS = (\text{KeyGen}, \mathcal{S}, \mathcal{V})$  be a digital signature scheme and let  $\mathcal{A}$  be adversary against the existentially unforgeability of  $DS$  playing the following game:*

**GAME.UF:**

Step 1.  $(\text{sk}, \text{vk}) \leftarrow \text{KeyGen}(1^k)$

Step 2. For  $i = 1, \dots, n$ :  $\{m_i \leftarrow \mathcal{A}(\text{vk}, m_1, \sigma_1, \dots, m_{i-1}, \sigma_{i-1}); \sigma_i \leftarrow \mathcal{S}_{\text{sk}}(m_i)\}$

Step 3.  $(\tilde{m}, \tilde{\sigma}) \leftarrow \mathcal{A}(\text{vk}, m_1, \sigma_1, \dots, m_n, \sigma_n)$

Here,  $n$  is of polynomial order of  $k$ . We define the advantage of the adversary  $\mathcal{A}$  as

$$\epsilon_{\mathcal{A}} = \Pr[\tilde{m} \notin \{m_1, \dots, m_n\} \text{ and } \mathcal{V}_{\text{vk}}(\tilde{m}, \tilde{\sigma}) = \text{“accept”}].$$

Moreover, we define  $\epsilon$  as  $\max_{\mathcal{A}}(\epsilon_{\mathcal{A}})$ , where the maximum is taken over all adversaries modeled as probabilistic polynomial time Turing machines. DS is said to be existentially unforgeability under adaptive chosen-message attacks (UF-CMA), if  $\epsilon$  is negligible in the security parameter  $k$ .

### 2.3.3 The Hash-Then-Sign Paradigm

When inspecting Example 2.3 (plain RSA signatures) in more detail, two problems catch the eye. First, this signature scheme seems to be quite weak regarding the above defined security notions. Even with a key-only attack existential forgeries are easily obtained just by choosing  $\sigma \in \mathbb{Z}_n$  at random, computing  $m = \sigma^e \bmod n$  and presenting  $\sigma$  as a forged signature on  $m$ . With a chosen-message attack, the adversary can sign any message of her choice due to the homomorphic properties of RSA: to sign  $m$ , choose  $r \in \mathbb{Z}_n$  at random and query the signing oracle on  $r^e m \bmod n$ . The signature  $m^d \bmod n$  is then easily obtained by multiplying the oracles response with  $r^{-1} \bmod n$ .

The second problem is the limited message space  $\mathbb{Z}_n$ . Signing of messages of arbitrary length might be desirable. A presumed solution of both problems is the use of *hash functions*. A hash function is an efficiently computable procedure that maps strings of arbitrary length to strings of fixed length, say 256 bits. The most important property of a hash functions is *collision resistance* (*i. e.*, it is infeasible to find two different strings mapping to the same value). In particular, it can easily be shown that—when ignoring pathological cases—collision resistance implies one-wayness (*i. e.*, given a randomly chosen hash value, it is infeasible to find a pre-image).

The well-known hash-then-sign technique is now described as follows: Instead of signing a message  $m$  directly, compute the hash  $h$  of  $m$  and sign the hash value:  $\sigma = \mathcal{S}_{\text{sk}}(h)$ . To verify if  $\sigma$  is a valid signature of  $m$ , compute the hash  $h$  of  $m$  and apply the regular verification algorithm to  $\sigma$  and  $h$ . This approach has several benefits. First, the efficiency is increased as hash functions in general are very fast, and the costly public-key signing operation is now required for short hash values only. This effect is two-edged, however, because a sparse message space may lead to new insecurities. Therefore, Bellare and Rogaway recommend to use a *full domain hash*, *i. e.*, a hash function that maps strings uniformly to the full domain of the signature scheme [BR93]. Second, messages of arbitrary length can be signed. Third, the above described trivial attacks against plain RSA signatures do not work any longer due to the one-wayness of the hash-function. Indeed, in the random oracle model, the full domain hash version of RSA can be proven to be EF-CMA.



## Chapter 3

# Improved and Explicit Security Reduction for RSA-OAEP and RSA-Paillier

In this chapter, we introduce a conceptually very simple and demonstrative algorithm for finding small solutions  $(x, y)$  of the linear congruence  $ax + y = c \pmod n$ , where  $\gcd(a, n) = 1$ . Our new algorithm is a variant of the Euclidian algorithm. Unlike former methods, it finds a small solution whenever such a solution exists. Further it runs in time  $\mathcal{O}((\log n)^3)$ , which is the same as the best known previous techniques, *e. g.*, lattice-based solutions.

We then apply our algorithm to RSA-OAEP and RSA-Paillier to obtain better security reduction proofs.

### 3.1 Introduction

Lattice reduction algorithms have been successfully applied to many branches of modern cryptography. In particular, this methods can be used to construct small solutions  $(x, y)$  of the linear modular congruence

$$ax + y = c \pmod n, \tag{3.1}$$

where the integers  $a$  and  $n$  are coprime, *i. e.*,  $\gcd(a, n) = 1$ .

In the random oracle model, the OAEP conversion is a technique to design a CCA secure encryption scheme from any trapdoor one-way permutation [BR94]. We write  $f$ -OAEP if  $f$  is the underlying trapdoor function. Today's most famous cryptosystem—RSA-OAEP—is a result of this work. By using lattice reduction for constructing small solutions of equation (3.1), Fujisaki *et al.* showed that provided the RSA assumption holds, RSA-OAEP is indistinguishable under adaptive chosen ciphertext attacks (IND-CCA) in the random oracle model [FOPS01]. Their paper is based on previous work of Bellare *et al.* [BR94] and Shoup [Sho02].

In the standard model, on the other hand, it is known that RSA-Paillier encryption scheme is indistinguishable under chosen plaintext attacks (IND-CPA). Following the work

of Sakurai *et al.* [ST02], Catalano *et al.* proved that the one-wayness of RSA-Paillier is equivalent to that of RSA by using the above described technique with  $c = 0$  [CGHGN01].

As motivated in Section 2.1.3, it is an important aim in cryptography to improve security reduction proofs, because the proposed size of the security parameters of a cryptosystem is directly influenced by the reduction costs.

In this chapter, we introduce a conceptually much simpler and demonstrative algorithm for finding small solutions of equation (3.1). Our new algorithm is a variant of the Euclidian algorithm. Unlike the lattice-based method, it exploits that the sought-after small solution is non-negative. Further, it runs in time  $\mathcal{O}((\log n)^3)$ , which is the same as the lattice-based method.

We then apply our algorithm to the security proof of RSA-OAEP to enhance the advantage of the reduction algorithm. The proof of RSA-OAEP is divided into two parts [FOPS01]. First the CCA security of the general OAEP conversion scheme under the so-called partial-domain one-wayness of the underlying trapdoor permutation is proved. Then, in the second part, the homomorphic properties of RSA function are exploited to show the equivalence of partial-domain one-wayness and full-domain one-wayness in the RSA case.

The second part, however, does not work for all values of  $a$  of equation (3.1). We call  $a$  a *bad value* if there is no warrant that the method operates successfully. In [FOPS01], Fujisaki *et al.* upperbound the number of bad values for  $a$  by the term  $2^{2k_0+6}$ , where  $k_0$  is the maximal bit-length of the sought-after small solution. Obviously, this result is not optimal, especially if the bound  $k_0$  is close to half of  $k$ , the bit-length of  $n$ . One reason for the non-optimal performance of the lattice-based method is that it does not exploit all the information given about the sought-after solution. Namely, it takes no advantage of the fact that the solution is non-negative, not only small in absolute value. Using our proposed algorithm, we are able to upper-bound the number of bad values for  $a$  by  $2^{2k_0+1-2\log\log k}$  instead of  $2^{2k_0+6}$ .

Finally, for RSA-Paillier, we use our new algorithm to construct an alternative reduction proof, extending the important work of Catalano *et al.* [CNS02]. Based on the analysis of our algorithm, we provide the exact security analysis while Catalano *et al.* gave asymptotic results only.

Although the achieved improvements are not dramatic ones, we assume that our proposed algorithm is a useful tool for analyzing appropriate security proofs rigorously. A general difference between the old approaches and our solution is that previous methods are intended to find one small solution only. If there exists only one small solution within the given bounds, then these methods succeed. Otherwise, the correct solution may be missed. Our proposed algorithm, in contrast, constructs a well-defined small solution, and, moreover, based on its analysis we are able to describe a method for finding *all* small solutions. The latter is efficient unless there are exponentially many small solutions. For cryptanalytic purposes, the known heuristic solutions work well enough, but in provable security the new algorithm might be preferable due to its simplicity and transparency.

### 3.1.1 Related Work

The task of finding small solutions for linear modular congruences is not a new one in cryptography. In 1985, De Jonge and Chaum developed an attack against some padding

techniques for RSA signature schemes [JC86], which was extended in 1997 by Girault and Misarsky [GM97]. These attacks utilize an affine variant of the Euclidian algorithm for solving two-variable linear modular equations with small solutions. But it has to be stressed that this algorithm may fail, even if a unique small solutions exists.

If  $c = 0$ , it is possible to find small solutions by means of continued fractions. Again, the Euclidian algorithm is used. But as before, this method is only heuristic, *i. e.*, it does not succeed with all input. We review this method in Section 3.3.3.

Our algorithm, on the contrary, works for arbitrary inputs.

## 3.2 Security Reduction Algorithms of RSA-OAEP and RSA-Paillier

In this section, we review the reduction proofs of the CCA security of RSA-OAEP and the one-wayness of RSA-Paillier. In both cases, we are confronted with the problem of finding small solutions of modular congruences. We sketch the existing solutions which utilize lattice reduction methods.

### 3.2.1 RSA-OAEP

The OAEP (*Optimal Asymmetric Encryption Padding*) conversion technique, introduced by Bellare and Rogaway in 1994 [BR94], is an encryption method which can be instantiated with any trapdoor one-way permutation. We write  $f$ -OAEP if  $f$  is the underlying trapdoor function. The further ingredients of OAEP are a 2 round Feistel network<sup>1</sup>, an appropriate padding, some randomization and two hash functions modeled as random oracles. The overhead compared to plain RSA only consists of some extremely fast hashing and xor operations. Bellare and Rogaway were able to prove that  $f$ -OAEP is plaintext aware under the one-wayness assumption of  $f$ . *Plaintext awareness* is a security property which originally was defined for the random oracle setting only. Recently, analogue notions for the standard model have been proposed [BP04]. Informally speaking, an encryption scheme is plaintext aware if there exists a so called plaintext-extractor who is able to simulate the answers of the decryption oracle on any ciphertext constructed by the adversary. The intuition behind this notion is that an adversary against a plaintext aware scheme is unable to construct valid ciphertexts in a way different from picking some message and encrypting it. This implies that a decryption oracle does not provide the adversary with additional power, because she does not gain any new information by querying it. Thus, it was believed that plaintext awareness establishes security against chosen-ciphertext attacks, and this is the way Bellare and Rogaway argued to substantiate IND-CCA security for  $f$ -OAEP.

Surprisingly, in 2001 Shoup detected a gap in the original security proof [Sho02]: The plaintext-extractor may fail in simulating the decryption oracle's answers on ciphertexts that have been designed by an adversary who knows any valid ciphertext (not produced by herself). But the challenge ciphertext given the adversary *is* a valid one, and thus, the original proof given by Bellare and Rogaway only provides *weakly* plaintext awareness, which means that the plaintext-extraction only works correctly if the adversary has no access to the decryption oracle after receipt of the challenge ciphertext. Consequently,

<sup>1</sup>Feistel networks are well-known in the context of symmetric block ciphers like DES.

their arguments only prove  $f$ -OAEP being semantically secure against chosen ciphertext attacks of *lunchtime type* (IND-CCA1) under the one-wayness assumption of  $f$ . Even worse, assuming the existence of a special kind of trapdoor one-way functions, Shoup proposed an adaptive chosen ciphertext attack against the OAEP scheme instantiated with a particular trapdoor one-way permutation. In addition, Shoup was able to show that trapdoor one-way functions with the demanded special property exist with high probability if trapdoor one-way functions exist at all. Consequently, Shoup's considerations make it very unlikely that the OAEP scheme can be proved to be IND-CCA2 under the one-wayness assumption of the underlying trapdoor permutation alone.

Fortunately, it took not long to fix the security proof. A close look at Shoup's attack reveals that the adversary has to *partly* invert the trapdoor function (*i. e.*, deduce  $s_1$  from  $f(s_1, s_2)$ ) in order to break the indistinguishability of the entire scheme. This observation leads to the definition of partial-domain one-wayness: A trapdoor permutation  $f$  is said to be *partial-domain one-way*, if it is hard to compute  $s_1$  from the knowledge of  $f(s_1, s_2)$ . Indeed, it was possible to prove the CCA security of the general OAEP conversion scheme under the partial-domain one-wayness of the underlying trapdoor permutation [FOPS01]. Note that partial-domain one-wayness is a stronger requirement than its full-domain pendant, hence, the corrected result is weaker than the original claim. But in the important special when instantiated with the RSA function, it turned out that due to the homomorphic properties of RSA partial-domain one-wayness implies full-domain one-wayness. Consequently, the original strong security result actually holds for RSA-OAEP. In the following, we sketch the latter proof:

First, we introduce some notations. If  $x$  is a natural number, we write  $[x]^l$  for the  $l$  most significant bits and  $[x]_l$  for the  $l$  least significant bits of the binary representation of  $x$ , respectively. Let  $n$  be a  $k$ -bit RSA modulus and fix  $k_0 < k/2$ . Suppose there is an algorithm  $\mathcal{I}$ —the partial inverter—that on the input  $C = m^e \bmod n$  returns the integer  $x := [m]^{k-k_0}$ . We show how to solve the RSA problem (compute  $m$  from  $C = m^e \bmod n$ ) using  $\mathcal{I}$  as a subroutine: Pick any  $a \in \mathbb{Z}_n^\times$  at random and run  $\mathcal{I}$  on the inputs  $C$  and  $C' := Ca^e \bmod n$ . Because of the homomorphic properties of the RSA function we know that  $C'$  is the encryption of  $ma \bmod n$ . Hence,  $\mathcal{I}$  determines the  $k - k_0$  most significant bits of  $m$  and  $ma \bmod n$ , respectively. We define  $u := [m]^{k-k_0}$ ,  $r := [m]_{k_0}$ ,  $v := [ma \bmod n]^{k-k_0}$  and  $s := [ma \bmod n]_{k_0}$ . Thus, we have

$$m = u \cdot 2^{k_0} + r \bmod n \quad \text{and} \quad ma = v \cdot 2^{k_0} + s \bmod n,$$

leading to

$$\begin{aligned} v \cdot 2^{k_0} + s &= a \cdot (u \cdot 2^{k_0} + r) \bmod n \\ \Rightarrow ar &= s + c \bmod n, \quad c = (v - ua) \cdot 2^{k_0} \bmod n. \end{aligned} \tag{3.2}$$

Solving this linear modular congruence yields the unknown value  $r = [m]_{k_0}$ , which completes the recovery of  $m$ .

However, the RSA-OAEP security proof is not tight. One problem lies in the random oracle part of the entire proof. In this proof, the partial inverter  $\mathcal{I}$  is constructed by using the IND-CCA adversary  $\mathcal{A}$  against RSA-OAEP as a subroutine. Unfortunately,  $\mathcal{I}$  does not determine  $u = [m]^{k-k_0}$  exactly; instead,  $\mathcal{I}$  constructs a set  $S$  with  $u \in S$ , where the magnitude of  $S$  equals the number of queries that  $\mathcal{A}$  sends to one of her random oracles. Thus, for each of the  $(\#S)^2$  possible combinations  $u, v$  taken from the output-sets  $S$  of

the two  $\mathcal{I}$ -runs, we get a linear modular congruence in the two unknowns  $r$  and  $s$ , where  $0 \leq r, s < 2^{k_0} < \sqrt{n}$ . Therefore, the reduction cost is quadratic in  $\#S$ . This is the main reason why the RSA-OAEP security proof is not meaningful for real-life parameters. Of course, an improvement of the congruence-solving-step will not affect this problem. Hence, it is an important future task to find a reduction proof where only one  $\mathcal{I}$ -run is needed.

In the following, we call  $x, y$  a *small* solution of the congruence (3.2) iff  $0 \leq x, y < 2^{k_0}$  holds. We explain how Fujisaki *et al.* find a small solution using the Gaussian reduction algorithm<sup>2</sup> (see also Figure 3.1). At first, compute a reduced basis  $(U, V)$  of the lattice

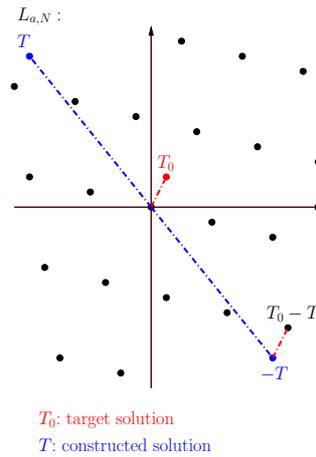


Figure 3.1: Lattice solution from Fujisaki *et al.* [FOPS01]

$L_{a,n} = \{(x, y) \in \mathbb{Z}^2 \mid ax = y \pmod n\}$  using the Gaussian algorithm. As we can easily find a sufficiently short basis of  $L_{a,n}$ —take, *e. g.*, the vectors  $(1, a)$  and  $(1, a + n)$ —this can be done in time  $\mathcal{O}((\log n)^3)$ . Let  $T_0$  be a small solution and  $T$  be any solution of eq. (3.2). To find  $T = (\tilde{x}, \tilde{y})$ , we can choose  $\tilde{x}$  as we like and then compute  $\tilde{y} = a\tilde{x} - c \pmod n$ . Define  $l = 2^{k_0+2}$  and assume that  $L_{a,n}$  is a so-called *l-good* lattice, meaning that there exists no non-zero lattice vector shorter than  $l$ . This choice of  $l$  together with the properties of a reduced basis guarantee two important facts: in the first place,  $T_0$  is unique as a small solution of eq. (3.2). Secondly, the coefficients of  $T_0$  in the basis  $(U, V)$  are smaller than  $1/2$  in absolute value. Thus, the coefficients (in  $(U, V)$ ) of the lattice point  $T_0 - T$  can be constructed simply by taking the closest integers to the coefficients of  $-T$ . This is a consequence of the uniqueness of basis representation. From knowledge of  $T$  and  $T_0 - T$ , we can easily construct  $T_0$ .

But as stated above, this method only works if the randomly chosen  $a$  yields an *l-good* lattice. Note that for fixed  $n$ ,  $L_{a,n}$  is uniquely determined even by a single vector  $v \in L_{a,n}$ . Thus, there are at most  $\pi 2^{2k_0+4} < 2^{2k_0+6}$  possible lattices containing a non-zero vector shorter than  $l = 2^{k_0+2}$ , and the number of bad values for  $a$  is bounded above by  $2^{2k_0+6}$ . Consequently, the probability of choosing a bad value is smaller than  $2^{2k_0+6-k}$ . Therefore, the total advantage of this reduction is greater than  $\varepsilon' = \varepsilon(\varepsilon - 2^{2k_0+6-k})$ , where  $\varepsilon$  denotes the advantage of the partial inverter  $\mathcal{I}$ . Note that  $\varepsilon'$  is non-negligible in  $k = \log n$  if  $\varepsilon$  is non-negligible in  $k$  and if  $k_0$  is adequate smaller than  $k$ , *i. e.*, if there is a rational number  $0 < t < 1/2$  such that  $k_0 < tk$ .

<sup>2</sup>Gaussian reduction can be regarded as a generalization of the Euclidian algorithm in dimension 2. For all results concerning lattice theory see [MG02].

### 3.2.2 RSA-Paillier

In Section 2.2, we introduced the Paillier cryptosystem from Eurocrypt 1999 (Example 2.2). In 2001, Catalano *et al.* proposed the variant RSA-Paillier to speed up the encryption process. Recall that the Paillier encryption of  $m \in \mathbb{Z}_n$  is computed as  $\mathcal{E}_{\text{pk}}(m) = (1 + mn)r^n \bmod n^2$ , where  $r \in \mathbb{Z}_n^\times$  is a random nonce. The idea of Catalano *et al.* was to replace the exponent  $n$  with an ordinary (small) RSA exponent  $e$ . Using a simple trick, decryption (different from original Paillier) is possible, too. The new scheme is faster in encryption, comparable in decryption, but unfortunately the homomorphic property breaks down.

In 2002, Sakurai and Takagi proved that RSA-Paillier is one-way if and only if the *Hensel-lifting problem* is hard [ST02], where the Hensel-lifting problem is defined as computing  $r^e \bmod n^2$  from a given ciphertext  $r^e \bmod n$  and an RSA public key  $(n, e)$ . Moreover, Sakurai and Takagi introduced a reduction algorithm for solving the RSA problem using the Hensel-lifting oracle as a subroutine, thus the one-wayness of RSA was reduced to the one-wayness of RSA-Paillier. But unfortunately, this algorithm was not particularly efficient (*i. e.*, for each bit of the secret message two oracle-calls were needed), and it could be proved to achieve a non-negligible advantage only in case of a perfect Hensel-lifting oracle. A short time later, Catalano *et al.* were able to show that the RSA problem could be solved by calling the (potentially non-perfect) Hensel-lifting oracle only twice [CNS02]. We shortly explain their technique in the following.

Assume that a random RSA ciphertext  $C = r^e \bmod n$  is given. We show how to reconstruct  $r$  given  $C, n, e$  using Hensel lifting. First, we obtain  $r^e \bmod n^2$  by invoking the Hensel lifting oracle. Then we query the Hensel lifting oracle on  $a^e r^e \bmod n$  for a randomly chosen integer  $a \in \mathbb{Z}_n^\times$ . We receive  $\mu^e \bmod n^2$ , where  $\mu = ar \bmod n$ . There is an integer  $z$  such that  $ar = \mu(1 + zn) \bmod n^2$  holds, and modulo  $n$  it can be computed from the equation  $a^e r^e = \mu^e(1 + ezn) \bmod n^2$ , because  $e, n, a, r^e \bmod n^2$ , and  $\mu^e \bmod n^2$  are known. Now consider the two-dimensional lattice  $L = \{(R, U) \in \mathbb{Z}^2 \mid aR = U(1 + zn) \bmod n^2\}$ . Using standard lattice reduction techniques we can find a vector  $(r', \mu') \in L \cap [1, \dots, n-1]^2$  in polynomial time of  $\log n$ . As the sought-after vector  $(r, \mu)$  is an element of  $L$ , too, we have the relationship  $r'\mu = r\mu' \bmod n^2$ . Moreover, due to the size constraints  $0 < r, r', \mu, \mu' < n$ , we conclude that in fact equality holds over the integers, *i. e.*,  $r'\mu = r\mu'$ . Thus—if *w. l. o. g.* we assume  $r > r', \mu > \mu'$ —we deduce that  $r$  and  $\mu$  are multiples of  $r'/\gcd(r', \mu')$  and  $\mu'/\gcd(r', \mu')$ , respectively, where in both cases the factor equals  $\gcd(r, \mu)$ . As with overwhelming probability this factor is sufficiently small, it can be found efficiently by an exhaustive search.

Catalano *et al.* showed that their method works in time polynomial in  $\log n$  with a non-negligible advantage, but they gave no concrete bounds.

## 3.3 The Proposed Reduction Algorithm

Let  $n$  be a natural number,  $0 < a < n$ ,  $0 \leq c < n$ , and  $\gcd(a, n) = 1$ . In this section, we develop the new algorithm `Lin_Cong` for finding small solutions of the two-variable linear modular congruence

$$ax = y + c \bmod n. \tag{3.3}$$

To be more concrete, we introduce an algorithm for finding so-called  $x$ -minimal solutions of eq. (3.3).

**DEFINITION 3.1** ( $x$ -MINIMAL SOLUTION) *The pair  $(\hat{x}, \hat{y}), 0 \leq \hat{x} < n, 0 \leq \hat{y} < B$  is called a  $x$ -minimal solution of eq. (3.3) with respect to the bound  $B, 0 < B < n$ , if  $(\hat{x}, \hat{y})$  possesses the following properties:*

1.  $a\hat{x} = \hat{y} + c \pmod n$ .
2.  $\hat{x}$  fulfills the  $x$ -minimality condition: *If  $(x_{alt}, y_{alt})$  is a solution of the congruence (3.3) where  $0 \leq y_{alt} < B$  holds, then we have  $\hat{x} \leq x_{alt}$ .*

Note that due to the condition  $\gcd(a, n) = 1$ , for each fixed  $B$  there is exactly one  $x$ -minimal solution of eq. (3.3) w. r. t.  $B$ .

As a second step, we will propose an efficient variant of the algorithm with complexity  $\mathcal{O}((\log n)^3)$ . One application of the new algorithm is to replace the lattice based methods used in the reduction proofs described above.

Algorithm 1 presents the outline of our proposed algorithm.

---

**Algorithm 1: Lin\_Cong (Outline)**

---

**Input:**  $a, c, n, B$ , where  $0 < a, B < n, 0 \leq c < n$ , and  $\gcd(a, n) = 1$

**Output:**  $\hat{x}, \hat{y}$  such that  $a\hat{x} = \hat{y} + c \pmod n$  and  $\hat{x} \geq 0$  is minimal with respect to the property  $0 \leq \hat{y} < B$

```

1  $a' \leftarrow a; c' \leftarrow c; n' \leftarrow n;$ 
2  $y' \leftarrow -c' \pmod{n'};$ 
3 while  $y' \geq B$  do
4    $(a', n') \leftarrow (-n' \pmod{a'}, a');$  /*parallel assignment */
5    $c' \leftarrow c' \pmod{n'}; y' \leftarrow -c' \pmod{n'};$ 
6  $\hat{y} \leftarrow y'; \hat{x} \leftarrow a^{-1}(\hat{y} + c) \pmod n;$ 
7 return  $(\hat{x}, \hat{y})$ 

```

---

Before proving its correctness formally, we state some technical remarks and describe the idea of the proposed algorithm.

First note that  $\gcd(a', n') = \gcd(a, n) = 1$  and  $a' < n'$  holds in any iteration. Therefore, we observe that  $a' = 0$  is only possible if the corresponding  $n'$  (the old value  $a'$  in the previous iteration) equals 1. If this were the case, in line 5 of this previous iteration we would have computed  $y' = 0$  and the algorithm would have terminated. Consequently, the assertion  $a' \leftarrow -n' \pmod{a'}$  is always defined.

Let  $(\hat{x}, \hat{y})$  be the unique  $x$ -minimal solution of eq. (3.3) w. r. t.  $B$ . We show that Algorithm 1 (Lin\_Cong (Outline)) on the inputs  $a, c, n, B$  returns  $(\hat{x}, \hat{y})$ . To be more precise, the algorithm finds  $\hat{y}$  and then computes the corresponding  $\hat{x} = a^{-1}(\hat{y} + c) \pmod n$ . The main idea of the algorithm is to reduce the original problem to a smaller instance and to iterate this process. This is done as follows: From  $a\hat{x} = \hat{y} + c \pmod n$  we deduce  $a\hat{x} = \hat{y} + c + kn$  for a suitable  $k \in \mathbb{Z}$ . Euclidian division yields  $n = aq + r$  with  $0 \leq r < a$  and a positive integer  $q$ . Hence, we have

$$\begin{aligned} a\hat{x} = \hat{y} + c + kn = \hat{y} + c + k(aq + r) &\Rightarrow -rk = \hat{y} + c + a(kq - \hat{x}) \\ &\Rightarrow -rk = \hat{y} + c \pmod a \end{aligned} \tag{3.4}$$

Therefore, we have constructed the new linear modular congruence (3.4) with the new module  $a$  in the role of  $n$  and the new factor  $-r = -n \bmod a$  in the role of  $a$ . A solution of congruence (3.4) is given by  $(k, \hat{y}) = (\frac{a\hat{x} - \hat{y} - c}{n}, \hat{y})$ . The crucial point is the observation that this solution is the  $x$ -minimal solution *w. r. t.*  $B$  of (3.4), which we will justify soon. We define the following sequences by iterating this process:

$$\begin{array}{llll} n_0 = n & a_0 = a & c_0 = c & x_0 = \hat{x} \\ n_{i+1} = a_i & a_{i+1} = -n_i \bmod a_i & c_{i+1} = c_i \bmod n_{i+1} & x_{i+1} = \frac{a_i x_i - \hat{y} - c_i}{n_i} \end{array}$$

Note that the first three columns exactly describe the corresponding sequences produced by the algorithm `Lin_Cong (Outline)`. For this reason, we denote by  $f_{\text{Lin\_Cong}}$  the transformation  $(n_i, a_i, c_i) \mapsto (n_{i+1}, a_{i+1}, c_{i+1})$ . Let us write  $\text{cong}_i$  for the linear modular congruence defined with the parameters  $a_i, c_i$  and  $n_i$  according to the  $i$ th iteration of algorithm `Lin_Cong (Outline)`<sup>3</sup>:

$$\text{cong}_i \equiv a_i x = y + c_i \bmod n_i.$$

Inductively, we conclude that the value  $x_i$  occurring in the last column leads to a solution  $(x_i, \hat{y})$  of  $\text{cong}_i$ . Moreover, we can deduce the following lemma:

**LEMMA 3.1** *Let  $(x_i, \hat{y})$  be the  $x$ -minimal solution of  $\text{cong}_i$  w.r.t.  $B$  and let  $x_i > 0$ . Then  $(x_{i+1}, \hat{y})$  is the  $x$ -minimal solution of  $\text{cong}_{i+1}$  w.r.t.  $B$ . In particular, the  $y$ -value of the current  $x$ -minimal solution w.r.t.  $B$  does not change during the transformation  $f_{\text{Lin\_Cong}}$ , as long as  $x_i$  is non-negative.*

**PROOF** First, we prove an auxiliary claim about a kind of “solution lifting”:

*Let  $(x, y)$  be a solution of  $\text{cong}_{i+1}$ . Then the pair  $(\frac{y+c_i+xn_i}{a_i}, y)$  is a solution of  $\text{cong}_i$ . If in addition  $0 \leq x, y < n_{i+1}$  holds then  $0 \leq \frac{y+c_i+xn_i}{a_i} \leq n_i$  is fulfilled.*

Note that the value  $a_i = n_{i-1}$  cannot be zero, since otherwise iteration  $(i+1)$  would not have been reached. First, we show  $\frac{y+c_i+xn_i}{a_i} \in \mathbb{Z}$ . The recursion formulas define  $n_{i+1} = a_i$  and  $c_{i+1} = c_i \bmod n_{i+1} = c_i \bmod a_i$ . Hence there is an integer  $l$  such that  $c_{i+1}$  equals  $c_i + la_i$ . As  $(x, y)$  is a solution of  $\text{cong}_{i+1}$  we conclude

$$a_i \mid y + c_{i+1} - xa_{i+1} = y + (c_i + la_i) - x(-n_i \bmod a_i) \Rightarrow a_i \mid y + c_i + xn_i$$

It follows immediately that  $(\frac{y+c_i+xn_i}{a_i}, y)$  is an integer solution of  $\text{cong}_i$ .

Now assume  $0 \leq x, y < n_{i+1} = a_i$ . We have

$$y + c_i + xn_i \leq y + c_i + (a_i - 1)n_i < a_i + n_i + (a_i - 1)n_i = a_i + a_i n_i$$

Therefore, we conclude  $\frac{y+c_i+xn_i}{a_i} < n_i + 1$ , which finishes the proof of the auxiliary claim.

Now we are ready to proof the main lemma: Assume that  $(x_{\text{alt}}, y_{\text{alt}})$  is a solution of  $\text{cong}_{i+1}$  where  $0 \leq x_{\text{alt}} < n_{i+1}$  and  $0 \leq y_{\text{alt}} < B$ . Our goal is to show  $x_{\text{alt}} \geq x_{i+1} \geq 0$ . First we prove that  $x_{i+1}$  is non-negative. Note that  $a_i > 0$  must hold because otherwise the iteration  $(i+1)$  of the while loop would not have been reached. From the definition of

<sup>3</sup>In particular, if the exit condition  $-c_j \bmod n_j < B$  holds, then  $\text{cong}_i$  should be undefined for  $i > j$ .

$x_{i+1} = \frac{a_i x_i - \hat{y} - c_i}{n_i}$  and the condition  $x_i > 0$ , we therefore conclude  $\hat{y} + c_i + x_{i+1} n_i = a_i x_i > 0$ . Assume  $x_{i+1} < 0$ . Hence, we have

$$\hat{y} + c_i > n_i \Rightarrow \hat{y} > n_i - c_i \geq -c_i \pmod{n_i}. \quad (3.5)$$

Since  $(0, -c_i \pmod{n_i})$  is a solution of  $\text{cong}_i$ , ineq. (3.5) contradicts the preconditions, namely  $(x_i, \hat{y})$  being the  $x$ -minimal solution of  $\text{cong}_i$  *w. r. t.*  $B$  and  $x_i > 0$ . Consequently, we must have  $x_{i+1} \geq 0$ .

From the auxiliary claim, we conclude that the pair

$$\left( \frac{y_{alt} + c_i + x_{alt} n_i}{a_i}, y_{alt} \right)$$

is a solution of  $\text{cong}_i$ , in particular, we have  $0 \leq \frac{y_{alt} + c_i + x_{alt} n_i}{a_i} \leq n_i$ . As  $(x_i, \hat{y})$  is the  $x$ -minimal solution of  $\text{cong}_i$  *w. r. t.*  $B$ , we deduce

$$\begin{aligned} \frac{y_{alt} + c_i + x_{alt} n_i}{a_i} - x_i &\geq 0 \\ \Rightarrow x_{alt} &\geq \frac{a_i x_i - y_{alt} - c_i}{n_i}. \end{aligned} \quad (3.6)$$

In the case of  $y_{alt} \leq \hat{y}$  this immediately leads to the desired result  $x_{alt} \geq x_{i+1} = \frac{a_i x_i - \hat{y} - c_i}{n_i}$ . Thus, we consider the case  $y_{alt} > \hat{y}$ . Looking at the difference between  $x_{i+1}$  and the righthand side of eq. (3.6), we observe

$$x_{i+1} - \frac{a_i x_i - y_{alt} - c_i}{n_i} = \frac{-\hat{y} + y_{alt}}{n_i} < \frac{B}{n_i} < 1. \quad (3.7)$$

Note that the last inequality must hold since in the case of  $n_i \leq B$  the iteration  $(i + 1)$  of the while loop would not have been reached. Therefore, we finally conclude  $x_{alt} \geq x_{i+1}$  from eq. (3.6), eq. (3.7) and the fact that both of  $x_{alt}$  and  $x_{i+1}$  are integers.  $\square$

Lemma 3.1 implies that with each iteration of the while loop, the transformation  $f_{\text{Lin\_Cong}}$  constructs a smaller instance of the problem of finding the  $x$ -minimal solution, since the sequence of the moduli  $n_i$  is strictly monotone decreasing. This problem, however, is trivial in the following case:

**DEFINITION 3.2 (ZERO-MINIMUM CONDITION)** *Let  $a, c, n, B$  be integers, where  $0 < a, B < n$ ,  $0 \leq c < n$ , and  $\text{gcd}(a, n) = 1$  hold. The congruence  $ax = y + c \pmod{n}$  satisfies the zero-minimum condition with respect to  $B$ , if  $-c \pmod{n} < B$  holds.*

In fact, it is an easy observation that the  $x$ -minimal solution of the congruence  $ax = y + c \pmod{n}$  *w. r. t.*  $B$  is given by the pair  $(0, -c \pmod{n})$  iff  $ax = y + c \pmod{n}$  satisfies the zero-minimum condition *w. r. t.*  $B$ . The aim of the algorithm **Lin\_Cong (Outline)** is to convert the original congruence into a congruence satisfying the zero-minimum condition *w. r. t.*  $B$ . This is done using the transformation  $f_{\text{Lin\_Cong}}$ , which does not affect the  $y$ -value of the current  $x$ -minimal solution *w. r. t.*  $B$ .

Indeed, we can prove the correctness of algorithm **Lin\_Cong (Outline)**:

**THEOREM 3.1** *Algorithm 1 (Lin\_Cong (Outline)) is correct, i. e., given integers  $a, c, n, B$ , where  $0 < a, B < n$ ,  $0 \leq c < n$ , and  $\gcd(a, n) = 1$  holds, the algorithm terminates and outputs the unique  $x$ -minimal solution  $(\hat{x}, \hat{y})$  of the congruence  $ax = y + c \pmod n$  with respect to the bound  $B$  (see Definition 3.1).*

**PROOF** Let  $y_i$  denote the  $y$ -value computed by the algorithm Lin\_Cong (Outline) in the  $i$ th iteration of the while loop. Note that per definition this value yields the solution  $(0, y_i)$  of  $\text{cong}_i$ . For each  $i = 0, 1, 2, \dots$  the following holds: Either  $\text{cong}_i$  satisfies the zero-minimum condition *w. r. t.*  $B$  and consequently  $(0, y_i)$  is the  $x$ -minimal solution of  $\text{cong}_i$  *w. r. t.*  $B$ . Or  $x_i$ , the  $x$ -value of the  $x$ -minimal solution of  $\text{cong}_i$ , is greater zero and Lemma 3.1 tells us that  $(x_{i+1}, \hat{y})$  equals the  $x$ -minimal solution of  $\text{cong}_{i+1}$  *w. r. t.*  $B$ . As the sequence of the moduli  $n_i$  is strictly monotone decreasing, there must be an  $i \geq 0$  such that  $\text{cong}_i$  satisfies the zero-minimum condition *w. r. t.*  $B$ . If this iteration is reached (*i. e.*, we have  $y_i < B$  for the first time), then  $(0, y_i) = (x_i, \hat{y})$  must hold because according to Lemma 3.1 we know that the  $y$ -value of the  $x$ -minimal solution *w. r. t.*  $B$  remained changed. Obviously, the  $x$ -value computed in line 6 is the correct one.  $\square$

### 3.3.1 Algorithm Lin\_Cong

Analyzing algorithm Lin\_Cong (Outline) we observe that the parallel assignment in line 4 describes a variant of the Euclidian algorithm (set  $(a, b) \leftarrow (-b \pmod a, a)$  instead of set  $(a, b) \leftarrow (b \pmod a, a)$  for  $a \leq b$ ). Obviously, the result remains the same, but unfortunately the variant is less efficient. In particular, in the worst case we need  $a - 1$  steps (to see this, try  $a = b - 1$ ), which is by far not fast enough. But some modifications may be helpful: A closer look at the recursion formula  $(a, b) \leftarrow (-b \pmod a, a)$  discloses that problems occur if  $b - a \ll a$  holds. In the subsequent steps, the difference  $b - a$  will be subtracted from  $a$  and  $b$  until the resulting  $a$  is smaller than  $b - a$ . This procedure may take too long time. Its result will be  $(a \pmod (b - a), b - k(b - a))$ , where  $k$  equals  $a \div (b - a)$  (here,  $x \div y$  denotes the Euclidian quotient of  $x$  and  $y$ ). Therefore, we gain a notable speedup by the following case differentiation:

if  $b - a \geq a$  then set  $(a, b) \leftarrow (-b \pmod a, a)$   
                   else set  $(a, b) \leftarrow (a \pmod b - a, b - k(b - a))$  with  $k = a \div (b - a)$ .

But we need to be a little careful if we wish to assign this idea to the original algorithm (with  $a'$  in the role of  $a$  and  $n'$  in the role of  $b$ ). In detail, we must not ignore a reduction of the value  $c'$  which would have occurred in line 5 of one of the skipped steps. A possible way out is to skip fewer steps, *i. e.*, we subtract  $n' - a'$  until the resulting  $a'$  is smaller than  $n' - a'$  or  $c'$  is greater than the resulting  $n'$ . We will see in a while that these modifications are good enough to yield a polynomial running time (in  $\log n$ ). But before doing so, we have to face a last problem: It is possible that the value  $\hat{y}$  we are seeking for would be computed in one of the skipped steps. Note that in each skipped step the value  $y' = -c' \pmod n'$  is reduced by the amount  $n' - a'$  (This is true because for  $c' < n'$ , we have  $-c' \pmod n' = n' - c'$ , and due to the considerations above, the value  $c'$  remains constant). Hence, if the resulting  $y'$  exceeds the bound  $B$ , all the “invisible” values  $y'$  computed during the skipped steps do so, too. This means that it is possible to miss the sought-after value  $\hat{y}$  only in the last while cycle before termination. So we avoid missing the correct  $\hat{y}$  by doing the following: If steps have been skipped during the last while cycle, add  $n' - a'$  to the current value  $y'$  until  $y' + k(n' - a')$  exceeds  $B$  for the first time. Then,

set  $\hat{y} = y' + (k - 1)(n' - a')$  and compute the corresponding  $\hat{x}$ -value as usual. Algorithm 2 is the result of these considerations.

---

**Algorithm 2: Lin\_Cong**


---

**Input:**  $a, c, n, B$ , where  $0 < a, B < n$ ,  $0 \leq c < n$ , and  $\gcd(a, n) = 1$

**Output:**  $\hat{x}, \hat{y}$  such that  $a\hat{x} = \hat{y} + c \pmod n$  and  $\hat{x} \geq 0$  is minimal with respect to the property  $0 \leq \hat{y} < B$

```

1   $a' \leftarrow a; c' \leftarrow c; n' \leftarrow n;$ 
2   $y' \leftarrow -c' \pmod{n'};$ 
3  while  $y' \geq B$  do
4       $\text{diff} \leftarrow n' - a';$ 
5      if  $\text{diff} < a'$  and  $\text{diff} < n' - c'$  then
6           $k \leftarrow \min\{a' \div \text{diff}, (n' - c') \div \text{diff}\};$ 
7           $(a', n') \leftarrow (a' - k \cdot \text{diff}, n' - k \cdot \text{diff});$  /*parallel assignment */
8           $\text{flag} \leftarrow 1$ 
9      else
10          $(a', n') \leftarrow (-n' \pmod{a'}, a');$  /*parallel assignment */
11          $\text{flag} \leftarrow 0$ 
12      $c' \leftarrow c' \pmod{n'}; y' \leftarrow -c' \pmod{n'}$ 
13 if  $\text{flag} = 1$  /*The sought-after value  $\hat{y}$  may have been missed */
14 then
15      $k \leftarrow \left\lceil \frac{B - y'}{\text{diff}} \right\rceil - 1; \hat{y} \leftarrow y' + k \cdot \text{diff}$ 
16 else
17      $\hat{y} \leftarrow y'$ 
18  $\hat{x} \leftarrow a^{-1}(\hat{y} + c) \pmod n;$ 
19 return  $(\hat{x}, \hat{y})$ 

```

---

We can prove the following theorem:

**THEOREM 3.2** a) *The complexity of Algorithm 2 (Lin\_Cong) is  $\mathcal{O}((\log n)^3)$ .*

b) *Let  $a, c, n, B$  be integers, where  $0 < a, B < n$ ,  $0 \leq c < n$ , and  $\gcd(a, n) = 1$  holds. Algorithm 2 (Lin\_Cong) on the inputs  $a, c, n, B$  finds the  $x$ -minimal solution  $(\hat{x}, \hat{y})$  of the congruence  $ax = y + c \pmod n$  with respect to the bound  $B$ . In particular, it finds a small solution upperbounded by  $B$  whenever such a small solution exists.*

**PROOF** From the discussion above, we conclude that on each input Algorithm Lin\_Cong computes the same output as its slower variant Lin\_Cong (Outline). Thus, the second part of the theorem is an immediate consequence of Theorem 3.1. It remains to show that Algorithm Lin\_Cong runs in polynomial time. We distinguish four cases

1. The condition in line 5 is not fulfilled due to  $n' - a' = \text{diff} \geq a'$ . Hence, the else-case in line 10 is entered. From  $n' \geq 2a'$  we deduce that the assignment  $n' \leftarrow a'$  at least halves the value of  $n'$ .
2. The condition in line 5 is not fulfilled due to  $n' - a' = \text{diff} \geq n' - c'$ . Hence, the

else-case in line 10 is entered and  $n'$  is assigned to  $a'$ . Because of  $a' \leq c'$  the reduction of  $c'$  modulo  $n'$ (=  $a'$ ) in line 12 at least halves the value of  $c'$ .

3. The condition in line 5 is fulfilled and the value  $k$  computed in line 6 equals  $a' \div (n' - a')$ . In this case, the assignment  $a' \leftarrow a' - k(n' - a')$  done in line 7 is equivalent to  $a' = a' \bmod (n' - a')$ . As we have  $n' - a' < a'$ , this assignment at least halves the value of  $a'$ .
4. The condition in line 5 is fulfilled and the value  $k$  computed in line 6 equals  $(n' - c) \div (n' - a')$ . The value of  $k$  is chosen in order to achieve that  $n' - (k + 1)(n' - a') \leq c'$  holds. Hence, the reduction of  $c'$  modulo  $n'$  in line 12 of the following while cycle at least halves the value of  $c'$ .

Summing up, we conclude that at least in each second while cycle at least one of the values  $a'$ ,  $c'$  and  $n'$  is at least halved. Note that the algorithm terminates if  $a' = 0$ ,  $c' = 0$  or  $n' = 1$  holds. Thus, the number of while cycles is bounded above by  $\log a + 2 \log c + \log n$ . Each step during the while loop can be done in  $\mathcal{O}((\log n)^2)$ . Consequently, the time complexity of the algorithm `Lin_Cong` is  $\mathcal{O}((\log n)^3)$ .  $\square$

### 3.3.2 Finding All Small Solutions

In this paragraph, we show that algorithm `Lin_Cong` can be modified to find *all* small solutions  $(x, y)$  of the linear modular congruence (3.3). Here, we call  $(x, y)$  a *small* solution, if  $0 \leq x, y < \sqrt{n}$  is satisfied. The time needed for computing all small solutions is  $\mathcal{O}((\log n)^3) + l\mathcal{O}(\log n)$ , where  $l$  is the number of small solutions. The most important observation is that there is a quite simple relationship between all the small solutions in the case of  $c = 0$ . The general case  $c \neq 0$  can be easily derived from the special case. We will see that in both cases all small solutions are located on the same line.

#### The Case $c = 0$

Let  $(x_0, y_0)$  and  $(x_1, y_1)$  be two different small solutions of

$$ax = y \bmod n, \quad \gcd(a, n) = 1, \quad (3.8)$$

*i. e.*, we have

$$ax_0 = y_0 \bmod n \text{ and } ax_1 = y_1 \bmod n,$$

leading to

$$x_0y_1 = x_1y_0 \bmod n.$$

Due to the size-constraints we deduce that this relationship even holds in  $\mathbb{Z}$ . Consequently, all small solutions are located on the same line through the origin. Hence, to get all of these, we simply have to compute all integer multiples  $(k\hat{x}, k\hat{y})$ ,  $k \in \mathbb{Z}^{\geq 0}$ ,  $k\hat{x} < \sqrt{n}$ ,  $k\hat{y} < \sqrt{n}$ , where  $(\hat{x}, \hat{y})$  is the smallest non-zero solution of eq. (3.8). This solution can be constructed using algorithm `Lin_Cong`. However, if we run Algorithm `Lin_Cong` on an input with  $c = 0$ , then it will terminate at once with the result  $(0, 0)$ . But we are seeking for a non-zero solution, hence, we exploit the relationship  $ax = y \bmod n \Leftrightarrow a(x - 1) = y - a \bmod n$ . Namely, we run `Lin_Cong` on the input  $(a, n - a, n, \sqrt{n})$ , get the result  $(x', y')$ , and return  $(\hat{x}, \hat{y}) := (x' + 1, y')$ . Theorem 3.1 shows that  $(\hat{x}, \hat{y})$  indeed yields the smallest non-zero solution of eq. (3.8).

### The Case $c \neq 0$

Let  $(\hat{x}, \hat{y})$  be the small solution computed by algorithm `Lin.Cong` on the input  $(a, c, n, \sqrt{n})$  and let  $(x_{alt}, y_{alt})$  be a different small solution. In particular, the difference  $(x_{alt} - \hat{x}, y_{alt} - \hat{y})$  is a non-zero solution of eq. (3.8). As  $\hat{x}$  is minimal, we know  $\hat{x} < x_{alt}$ . Thus, we conclude  $0 < x_{alt} - \hat{x} < \sqrt{n}$ ,  $-\sqrt{n} < y_{alt} - \hat{y} < \sqrt{n}$ . We distinguish two cases:

1. If  $y_{alt} > \hat{y}$  holds, then  $(x_{alt} - \hat{x}, y_{alt} - \hat{y})$  is a small solution of congruence (3.8) and can be found as described above.
2. Otherwise,  $(x_{alt} - \hat{x}, y_{alt} - \hat{y})$  is a solution of congruence (3.8), too, but only small in absolute value (with a negative  $y$ -component). It is easy to see that we can find all solutions  $(x, y)$ ,  $0 \leq x < \sqrt{n}$ ,  $-\sqrt{n} < y \leq 0$  of congruence (3.8) by computing all small solutions of  $(-a)x = y \pmod n$  as usual and then altering the signs of the  $y$ -components.

Note that at most one of these two cases may appear: Assume there are two additional small solutions  $(x_{alt1}, y_{alt1})$  and  $(x_{alt2}, y_{alt2})$  with  $y_{alt1} > \hat{y}$  and  $y_{alt2} < \hat{y}$ . Then, the three differences  $(x_{alt1} - \hat{x}, y_{alt1} - \hat{y})$ ,  $(x_{alt2} - \hat{x}, y_{alt2} - \hat{y})$ , and  $(x_{alt1} - x_{alt2}, y_{alt1} - y_{alt2})$  form a non-degenerate triangular. This contradicts the fact that they are located on the same line through the origin.

Moreover, we can easily verify that for any solution  $(\hat{x}, \hat{y})$  of the congruence (3.3) and for any solution  $(x, y)$  of the corresponding congruence (3.8), the pair  $(\hat{x} + kx, \hat{y} + ky)$  also forms a solution of (3.3) for any integer  $k \in \mathbb{Z}$ .

All these observations lead to Algorithm 3.

In line 2, we use algorithm `Lin.Cong` to compute the small solution with the minimal  $x$ -coordinate  $\hat{x}$ . If even  $\hat{x}$  exceeds the bound  $\sqrt{n}$ , then obviously no small solution exists at all. The pair  $(x_0, y_0)$  computed in line 6 equals the smallest non-zero solution of congruence (3.8). As we have seen above, each sum of  $(\hat{x}, \hat{y})$  and an integer multiple of  $(x_0, y_0)$  yields a solution of  $ax = y + c \pmod n$ . But as  $\hat{x}$  is minimal, we only have to consider factors  $k \geq 1$ . If there is at least one small solution  $(\hat{x} + kx_0, \hat{y} + ky_0)$ , we know that all small solutions have to be of this shape. Hence, the while loop from line 8 to line 10 finds all remaining small solutions and the algorithm terminates in line 11. Otherwise we compute the smallest non-zero solution of  $ax = y \pmod n$  with a negative  $y$ -component (lines 12 and 13) and proceed in the same way as before.

Summing up, we have the following theorem:

**THEOREM 3.3** *Let  $a, c, n$  be integers, where  $0 < a, c < n$ , and  $\gcd(a, n) = 1$  holds. Algorithm 3 (`Lin.Cong.All`) finds all solutions  $(x, y)$ ,  $0 \leq x, y < \sqrt{n}$  of the linear modular congruence  $ax = y + c \pmod n$  in time  $\mathcal{O}((\log n)^3) + l\mathcal{O}(\log n)$ , where  $l$  is the number of these solutions.*

### 3.3.3 Comparison with the Continuous Fraction Method

Another frequently used method for finding small solutions of linear modular congruence with vanishing affine coefficient  $c$  is obtained by the continued fraction expansion. We call this method the *Euclidean reduction* (see [HW79] for the comprehensive treatment). To

**Algorithm 3:** Lin\_Cong\_All

---

**Input:**  $a, c, n$ , where  $0 < a, c < n$ , and  $\gcd(a, n) = 1$   
**Output:**  $S = \{(x, y) \mid ax = y + c \pmod n, 0 \leq x, y < \sqrt{n}\}$

- 1  $S \leftarrow \{\}$ ;
- 2  $(\hat{x}, \hat{y}) \leftarrow \text{Lin\_Cong}(a, c, n, \sqrt{n})$ ;
- 3 **if**  $\hat{x} \geq \sqrt{n}$  **then** return  $S$  and halt;
- 4 append  $(\hat{x}, \hat{y})$  to  $S$ ;
- 5  $(x', y') \leftarrow \text{Lin\_Cong}(a, n - a, n, \sqrt{n})$ ;
- 6  $(x_0, y_0) \leftarrow (x' + 1, y')$ ;
- 7  $k \leftarrow 1$ ;
- 8 **while**  $\hat{x} + kx_0 < \sqrt{n}$  and  $\hat{y} + ky_0 < \sqrt{n}$  **do**
- 9     append  $(\hat{x} + kx_0, \hat{y} + ky_0)$  to  $S$ ;
- 10     $k \leftarrow k + 1$
- 11 **if**  $\#S > 1$  **then** return  $S$  and halt;
- 12  $(x', y') \leftarrow \text{Lin\_Cong}(n - a, a, n, \sqrt{n})$ ;
- 13  $(x_0, y_0) \leftarrow (x' + 1, -y')$ ;
- 14  $k \leftarrow 1$ ;
- 15 **while**  $\hat{x} + kx_0 < \sqrt{n}$  and  $\hat{y} + ky_0 \geq 0$  **do**
- 16     append  $(\hat{x} + kx_0, \hat{y} + ky_0)$  to  $S$ ;
- 17     $k \leftarrow k + 1$
- 18 **return**  $S$

---

resume, this method finds all fractions  $p/q$  nearby a rational number  $\alpha$  (*i. e.*, we have  $|\alpha - p/q| < 1/(2q^2)$ ), where the fractions  $p/q$  come in their lowest terms. Assume that we want to find small solutions that do not exceed  $\sqrt{n}$  of the congruence  $ax = y \pmod n$ , where  $\gcd(a, n) = 1$  holds. As we have already shown in Section 3.3.2, all these solutions are located on the same line through the origin. Therefore, there exists a solution  $(\hat{x}, \hat{y})$  such that  $\gcd(\hat{x}, \hat{y}) = 1$  is fulfilled. From  $a\hat{x} = \hat{y} \pmod n$  we conclude that there is an integer  $k$  such that  $a\hat{x} = \hat{y} + kn$ . We have

$$\frac{a}{n} - \frac{k}{\hat{x}} = \frac{\hat{y}}{n\hat{x}}. \quad (3.9)$$

If  $2\hat{x}\hat{y} < n$  holds, the upper-bound of  $\hat{y}/n\hat{x}$  is  $1/2\hat{x}^2$ . If in addition the rational number  $k/\hat{x}$  is irreducible, *i. e.*,  $\gcd(k, \hat{x}) = 1$ , we can find the integer  $\hat{x}$  and thus  $\hat{y}$  by using the Euclidian reduction method with  $\alpha = a/n$ . Note that  $\gcd(k, \hat{x}) = 1$  holds because  $\gcd(\hat{x}, \hat{y}) = 1$  is satisfied: If  $\gcd(k, \hat{x}) = 1$  were not true, there were an integer  $\delta > 1$  such that  $\gcd(k, \hat{x}) = \delta$ . From  $a\hat{x} - nk = \hat{y}$ , we had  $\delta \mid \hat{y}$  and hence,  $\delta \mid \gcd(\hat{x}, \hat{y})$ , contradicting  $\gcd(\hat{x}, \hat{y}) = 1$ .

Summing up, we can use this method if we know that the product  $2\hat{x}\hat{y}$  does not exceed  $n$ . Since the asymptotic computational complexity is the same, we prefer the use of algorithm Lin\_Cong, which finds  $\hat{x}, \hat{y}$ , even if  $2\hat{x}\hat{y} < n$  is not fulfilled.

### 3.4 Security Reduction Analysis using the Proposed Algorithm

In this section, we show how Algorithm 2 (Lin\_Cong) may be applied to the reduction proofs of RSA-OAEP and RSA-Paillier. In the case of RSA-OAEP, we will upper-bound the number of bad values  $a$  by  $2^{2k_0+1-2\log\log k}$ , compared to the former bound  $2^{2k_0+6}$ . Regarding to RSA-Paillier, we will give an explicit reduction algorithm based on the work of Catalano *et al.* [CNS02]. We will achieve reduction time  $2t + \mathcal{O}((\log n)^3 \varepsilon^{-2})$  and advantage  $\varepsilon' > \varepsilon^2/5$ , where  $t$  and  $\varepsilon$  are the time and the advantage of the Hensel-lifting oracle, respectively.

#### 3.4.1 Application to RSA-OAEP

In Section 3.2.1, we have described the reduction proof given by Fujisaki *et al.* [FOPS01]. Remember that they have constructed the following congruence

$$ax = y + c \pmod n, \quad c = (v - ua) \cdot 2^{k_0} \pmod n, \quad (3.10)$$

where  $u$  and  $v$  are built of the  $k - k_0$  most significant bits of  $m$  or  $ma \pmod n$ , respectively. In the RSA-OAEP case we call  $(x, y)$  a *small* solution of the congruence (3.10) iff  $0 \leq x, y < 2^{k_0}$  holds. The congruence (3.10) is known to have the small solution  $(r, s)$ , where  $r$  is built from the remaining  $k_0$  least significant bits of  $m$ .

In Section 3.2.1, we have already seen that the lattice based method only works if the randomly chosen value  $a$  yields an  $l$ -good lattice. In contrast, algorithm Lin\_Cong always finds a small solution, provided a small solution exists at all. But it has to be stressed, that referring to the lattice method the choice of a good value  $a$  ensures that there exists exactly one small solution. This is an important property, because if the small solution  $(r, s)$  is not unique, there is of course no warrant that the solution computed with our algorithm is the correct one. A possible way out is to use algorithm Lin\_Cong\_All instead, which computes all small solutions, and to test each of them. But this is only efficient if the set of small solutions is not too big. Let  $l$  be a natural number. We want to upperbound the probability that the number of small solutions does not exceed  $l$ . As we have seen in Section 3.3.2, each small solution is of the shape  $(\hat{x} + kx_0, \hat{y} + ky_0)$ , where  $(\hat{x}, \hat{y})$  is the  $x$ -minimal solution computed by the algorithm Lin\_Cong and  $(x_0, y_0)$  is either the shortest vector of  $\{(x, y) \mid ax = y \pmod n, 0 < x, y < 2^{k_0}\}$  or it is the shortest vector of  $\{(x, y) \mid ax = y \pmod n, 0 < x < 2^{k_0}, -2^{k_0} < y < 0\}$ . Hence, there are at most  $l$  small solutions of equation (3.10), iff the congruence  $ax = y \pmod n$  has no solution  $(x, y)$ , where

$$0 < x < 2^{k_0}/l, \quad -2^{k_0}/l < y < 2^{k_0}/l. \quad (3.11)$$

We call  $a$  a *bad value*, if there exists a solution of  $ax = y \pmod n$  fulfilling the size constraints (3.11). If (3.11) holds for  $(x, y)$ , then there is exactly one  $a$  such that  $(x, y)$  is a solution of  $ax = y \pmod n$ , namely  $a = x^{-1}y \pmod n$ . Note that due to the size constraints no problems of computing modular inverses occur. Hence there are at most  $2^{2k_0+1}/l^2$  bad values of  $a$ . The maximal number is  $2^{2k_0+1}$  for  $l = 1$ .

Therefore, in case of using the lattice solution the probability to choose a bad value  $a$  is at least  $2^5$  times greater compared with the corresponding probability in case of

using algorithm `Lin_Cong_All`. We finish with the following theorem that is an immediate consequence of Theorem 3.3 and the considerations above:

**THEOREM 3.4** *Assume there is an adversary that on input  $n, e, m^e \bmod n$  returns the  $k - k_0$  most significant bits of  $m$  with advantage  $\varepsilon$  and in time  $t$ , where where  $2k_0 < k$  holds and  $(n, e)$  is an RSA public key with  $|n|_2 = k$ . Let  $l \leq (\log n)^2$  be any natural number. Then with advantage  $\varepsilon' > \varepsilon(\varepsilon - 2^{2k_0+1-k}/l^2)$  and in time  $2t + \mathcal{O}((\log n)^3)$  we can compute a set  $S$  with  $m \in S$  and  $\#S \leq l$ .*

If we set  $l = \log \log n$ , then it is still possible to check all values  $r \in S$  in time  $\tilde{\mathcal{O}}((\log n)^3)$  and we obtain the following corollary<sup>4</sup>:

**COROLLARY 3.1** *Assume there is an adversary that on input  $n, e, m^e \bmod n$  returns the  $k - k_0$  most significant bits of  $m$  with advantage  $\varepsilon$  and in time  $t$ , where  $2k_0 < k$  holds and  $(n, e)$  is an RSA public key with  $|n|_2 = k$ . Then we can break the RSA problem related to  $(n, e)$  with advantage at least  $\varepsilon(\varepsilon - 2^{2k_0+1-2\log \log k-k})$  and in time  $2t + \tilde{\mathcal{O}}((\log n)^3)$ .*

Note that this achievement is the more valuable, the smaller the difference  $k - 2k_0$  is. In the case of PKCS #1 v2.0, however,  $k_0$  is much smaller than  $k/2$ . Therefore in this case, the result is mainly of theoretical interest.

### 3.4.2 Application to RSA-Paillier

In Section 3.2.2, we have described the reduction proof given by Catalano *et al.* [CNS02]. Recall that they have constructed the following congruence

$$Ax = y \bmod n^2, \quad A = a(1 + zn)^{-1} = a(1 - zn) \bmod n^2, \quad (3.12)$$

which is known to have the solution  $(r, \mu)$ , where  $r, \mu$  are elements of  $\mathbb{Z}_n$  and  $r$  is the sought-after RSA message. Hence,  $(r, \mu)$  is a small solution of eq. (3.12) and can be constructed as described in Section 3.3.2, where we have seen how to find all small solutions. To be concrete, Algorithm 2 (`Lin_Cong`) on the input  $(A, n^2 - A, n^2, n)$  finds the smallest non-zero solution of the congruence (3.12) and all other small solutions come as integer multiples of this special solution.

Hence, Algorithm 4 describes an efficient method to solve the RSA problem with the aid of a Hensel-lifting oracle. From the results of Catalano *et al.* [CNS02] and Sakurai/Takagi [ST02] discussed in Section 3.2.2, we conclude that this algorithm also provides a reduction from the RSA problem to the one-wayness of the RSA-Paillier cryptosystem.

Obviously, the running time of this algorithm is  $\mathcal{O}((\log n)^3)$  plus the time needed for calling the Hensel-lifting oracle twice. To receive the original value  $r$ , we have to test if  $(kr)^e = c \bmod n$  holds, where the multiplier  $k$  runs from 1 to the (unknown) number  $\gcd(r, \mu), \mu = ar \bmod n$ . In the following, we upper-bound the probability that  $\gcd(r, \mu)$  is not sufficiently small. We exploit the following estimate (see [NZ76]):

$$\frac{\pi^2}{3} \left( \frac{2n^2 - 2n}{4n^2 + 4n + 1} \right) < \sum_{i=1}^n \frac{1}{i^2} < \frac{\pi^2}{3} \left( \frac{2n^2 + 2n}{4n^2 + 4n + 1} \right).$$

<sup>4</sup>Here,  $\tilde{\mathcal{O}}$  denotes the soft  $\mathcal{O}$  notation, which ignores logarithmic factors. As we have  $\log \log n = o(\log n)$ , this measurement is reasonable for practical purposes.

**Algorithm 4:** OW\_RSA\_Paillier**Input:**  $(n, e)$  RSA public key,  $c$  RSA ciphertext,  $\mathcal{O}_{Hensel}$  Hensel-lifting oracle**Output:** Message  $r < n$  such that  $c = r^e \pmod n$  or an integer divisor of  $r$ 


---

```

1  $t \leftarrow \mathcal{O}_{Hensel}(c)$ ;
2  $a \leftarrow \mathbb{Z}_n^\times$  /*  $a$  is chosen uniformly at random from  $\mathbb{Z}_n^\times$  */
3  $s \leftarrow \mathcal{O}_{Hensel}(a^e c \pmod n)$ ;
4  $v \leftarrow ta^e s^{-1} \pmod{n^2}$ ;
5  $z \leftarrow \frac{(v-1)}{n} e^{-1} \pmod n$ ;
6  $A \leftarrow a(1 - zn) \pmod{n^2}$ ;
7  $(\hat{x}, \hat{y}) \leftarrow \text{Lin\_Cong}(A, n^2 - A, n^2, n)$ ;
8 return  $\hat{x} + 1$ 

```

---

Hence, we have

$$\begin{aligned}
\#\{(a, b) \in [1, \dots, n]^2 \mid \gcd(a, b) > B\} &< \sum_{i=B+1}^n \frac{n^2}{i^2} \\
&< \frac{n^2 \pi^2}{3} \left( \frac{2n^2 + 2n}{4n^2 + 4n + 1} - \frac{2B^2 - 2B}{4B^2 + 4B + 1} \right) \\
&< \frac{n^2 \pi^2}{3} \left( \frac{1}{2} - \frac{2B^2 - 2B}{4B^2 + 4B + 1} \right).
\end{aligned}$$

A simple computation shows

$$\frac{1}{2} - \frac{2B^2 - 2B}{4B^2 + 4B + 1} < \frac{1}{B}.$$

Therefore, we finally conclude

$$\#\{(a, b) \in [1, \dots, n]^2 \mid \gcd(a, b) > B\} < \frac{4n^2}{B}.$$

The values  $r$  and  $\mu$  are independently chosen and uniformly distributed elements of  $\mathbb{Z}_n$ . Replacing  $5\varepsilon^{-2}$  for  $B$ , we therefore deduce that the probability that  $\gcd(r, \mu)$  exceeds  $5\varepsilon^{-2}$  is bounded above by  $4\varepsilon^2/5$ .

This leads to the following theorem:

**THEOREM 3.5** *Let  $(n, e)$  be an RSA public key and let  $\mathcal{O}_{Hensel}$  be the Hensel-lifting oracle that computes  $r^e \pmod{n^2}$  for given  $r^e \pmod n$  with advantage  $\varepsilon$  and in time  $t$ . Using  $\mathcal{O}_{Hensel}$  as a subroutine, we can break the RSA problem with advantage  $\varepsilon' > \varepsilon^2/5$  and in time  $2t + \mathcal{O}((\log n)^3 \varepsilon^{-2})$ .*

**EXAMPLE 3.1 (REDUCTION ALGORITHM OW\_RSA\_Paillier)** We finish with a small example of reduction algorithm OW\_RSA\_Paillier. We choose the public key of the RSA-Paillier cryptosystem as  $(n, e) = (9359629, 7)$ . In our case,  $n^2$  equals 87602655017641. Let  $C = 2592708$  be the target ciphertext. We intend to find the integer  $r$  such that  $C = r^e \pmod n$  using the oracle  $\mathcal{O}_{Hensel}$ .

In line 1, we query  $C$  to oracle  $\mathcal{O}_{Hensel}$ , and we obtain  $t = \mathcal{O}_{Hensel}(C) = 37278188147938$ . In line 2, a random integer  $a \in \mathbb{Z}_n^\times$  is generated, and we choose  $a = 5973500$ . In line 3, we compute  $\mu^e = a^e C \bmod n$ , ask it to oracle  $\mathcal{O}_{Hensel}(\mu^e \bmod n)$ , and we obtain  $\mu^e \bmod n^2 = 59913274976876$ . In lines 4 and 5, an integer  $z$  with  $ar = \mu(1 + zn) \bmod n^2$  is computed, and in our case, we calculate  $z = 9040417$ . In line 6 we obtain the linear equation  $Ar = \mu \bmod n^2$  for  $A = 35049167803493$  and two unknown variables  $0 < r, \mu < n$ .

In the following, we solve this linear equation using Algorithm 2 (Lin\_Cong). We list the intermediate values of  $n', a', c'$  and  $y'$ , where  $n', a'$  and  $c'$  are initialized with  $n^2, A$ , and  $n^2 - A$ , respectively. The while loop terminates if  $y' < n$  holds. Algorithm Lin\_Cong computes values  $\hat{x}$  and  $\hat{y}$ , which in our case equal  $r - 1$  and  $\mu$ .

$n'$	$a'$	$c'$	$y'$	
87602655017641	35049167803493	52553487214148	35049167803493	
35049167803493	17544848392838	17504319410655	17544848392838	
17544848392838	40528982183	17504319410655	40528982183	
40528982183	4200892401	36328089782	4200892401	
4200892401	1479941827	2720950574	1479941827	
1479941827	238933080	1241008747	238933080	
238933080	192589733	46343347	192589733	
53559692	7216345	46343347	7216345	← loop exit

The output values are  $(1835097, 7216345) = (r - 1, \mu)$ . In particular, we successfully find  $r = 1835098$ .

### 3.5 Conclusion

In this chapter, we revisited the security reduction algorithms related to the RSA-OAEP and the RSA-Paillier cryptosystems. These algorithms exploit techniques of finding small solutions of linear modular equations. The standard algorithms for solving this task are lattice-based (Gaussian reduction) or based on continued fractions (Euclidean reduction). We proposed an efficient alternative algorithm and showed its preferences. In the case of RSA-OAEP, we were able to enhance the advantage of the reduction proof. For RSA-Paillier, the previous solution gave qualitative results only (*i. e.*, existence of a polynomial time reduction with non-negligible advantage). The use of our new algorithm, in contrast, provides the complete security reduction proof, including explicit bounds for time costs and the achieved advantage.

## Chapter 4

# A New Rabin-type Trapdoor One-way Function and Its Applications

Public key cryptography has been invented to overcome some key management problems in open networks. Although nearly all aspects of public key cryptography rely on the existence of (trapdoor) one-way functions, only a very few candidates of this primitive have been observed yet. In this chapter, we introduce a new trapdoor one-way function based on the hardness of factoring integers of  $p^2q$ -type. We point out similarities between the proposed function and Rabin-type modular squaring. Most interestingly, two novel trapdoor one-way permutations can be derived from our approach. Moreover, we develop several applications to homomorphic encryption, hybrid encryption, fail-stop signature schemes and trapdoor commitments.

### 4.1 Introduction

Informally, a *one-way function* is a function that is “easy” to compute but “hard” to invert. If there exists some token of information that makes the inversion also an easy task, then we call the function *trapdoor*. Trapdoor one-way functions (in particular the bijective trapdoor one-way permutations) are used as building blocks for various kind of cryptographic schemes, *e. g.*, asymmetric encryption, digital signatures, and private information retrieval. There is no doubt that the concept of trapdoor one-way functions is of particular importance especially in public key cryptography. Nevertheless, just a relatively small number of promising candidates can be found in the literature. Promising here means that a presumed hard problem such as the factorization of large integers can be reduced to the one-wayness of the trapdoor function in question. As not even the pure existence of one-way functions can be proved today<sup>1</sup>, this kind of *provably secure* trapdoor one-way functions is the best alternative solution at present.

---

<sup>1</sup>Interestingly, the current knowledge in complexity theory does not even allow to prove the existence of one-way functions assuming  $\mathcal{P} \neq \mathcal{NP}$ . On the other hand, it is known that the existence of one-way functions implies  $\mathcal{P} \neq \mathcal{NP}$ .

### 4.1.1 Previous Work

The oldest and still best known candidate trapdoor one-way permutation is the RSA function, invented 1971 by Rivest, Shamir and Adleman (see Example 2.1). RSA is defined as modular exponentiation with exponents coprime to the order of the multiplicative residue group [RSA78]. The factors of the modulus can serve as a trapdoor to invert the RSA function, but the opposite direction is unknown. Thus, RSA is not provably equivalent to factoring, and there are serious doubts that this equivalence holds indeed [BV98]. Anyway, as the RSA problem has been extensively studied for decades, nowadays inverting the RSA function is widely accepted as a hard problem itself. Slightly later, Rabin observed that the special case of modular squaring leads to the desired equivalence to factoring [Rab79]. Modular squaring, however, is not a permutation, it is four-to-one (in case of a two-factor modulus). This drawback can be overcome: squaring modulo a Blum integer<sup>2</sup>  $n$  is a permutation of  $\text{QR}(n)$ , where  $\text{QR}(n) := \{x \in \mathbb{Z}_n^\times \mid \exists y : y^2 = x \pmod n\}$  denotes the group of *quadratic residues* modulo  $n$ . The resulting trapdoor permutation is referred to as Blum-Williams function in the literature, and an extension (exponent  $2e$ , where  $e$  is coprime to  $\lambda(n)$ ) is denoted Rabin-Williams function. More factorization-based trapdoor permutations were proposed by Kurosawa *et al.* [KIT88], Paillier [Pai99a, Pai99b], and Galindo *et al.* [GMMV03]. A survey on trapdoor permutations including some less established candidates can be found in [PG97].

### 4.1.2 Our Contributions

In this chapter, we introduce a rather simple trapdoor one-way function equivalent to factoring integers of the shape  $n = p^2q$ . As many previous candidates, our proposed trapdoor function is also a variant of the RSA function, namely in our case the public exponent is the same as the modulus  $n = p^2q$ . On the domain  $\mathbb{Z}_n^\times$ , the function  $x \mapsto x^n \pmod n$  is  $p$ -to-one, but restricted to the subgroup of  $n$ -th residues modulo  $n$ , it is indeed a permutation. These properties are similar to those of the Blum-Williams permutation (where  $n$ -th residues are replaced by quadratic residues). Analogical to the quadratic residuosity assumption, we assume that without knowledge of the factorization of  $n$ , it is hard to distinguish  $n$ -th residues from non-residues, whereas it is efficient if the factors of  $n$  are known. However, the restricted domain has some shortcomings, which also apply to Blum-Williams and Rabin-Williams functions: in practical applications, the data has to be preprocessed into the set of  $n$ -th, resp. quadratic residues. Supposably, this is one reason why the RSA function (with domain  $\mathbb{Z}_n$ ) is by far more widespread in commercial applications than Rabin-type functions. But fortunately, we can prove that for  $n$  of  $p^2q$ -type the set of  $n$ -th residues is isomorphic to  $\mathbb{Z}_{pq}^\times$ , thus our proposed trapdoor function also provides a bijection between the easy-to-handle domain  $\mathbb{Z}_{pq}^\times$  and the set of  $n$ -th residues. No such property is known for Rabin-type functions. Consequently, our trapdoor permutation can be used to encrypt *arbitrary* strings like keys. We provide an application in Section 4.4. Further advantages of our trapdoor one-way function in contrast to Rabin-type modular squaring accrue from the fact that the magnitude of the kernel is larger. This provides a higher degree of freedom in finding pre-images, and therefore offers the construction of fail-stop signatures (see Section 4.5) and trapdoor commitments (see Section 4.6). Moreover, we can extend our analysis to a Paillier-like function operating in the group  $\mathbb{Z}_{n^2}^\times$  for  $n$  of

<sup>2</sup>A *Blum integer* is a product of two distinct primes each congruent to 3 modulo 4.

$p^2q$ -type. Interestingly, it turns out that in this case we are able to reduce factoring to the one-wayness of Paillier's encryption scheme, whilst the one-wayness of Paillier's original scheme relies on a non-standard assumption only (see Section 4.3).

As in this chapter we give numerous applications to rather different cryptographic domains, more detailed motivations and full descriptions of our contributions in the particular fields are postponed to the respective sections.

## 4.2 A Trapdoor One-way Permutation Equivalent to Factoring

In this section, we introduce a new trapdoor one-way function and two associated trapdoor one-way permutations. We also give a short account on the mathematical background in order to deepen the understanding about the special properties of the group  $\mathbb{Z}_n^\times$  for  $n$  of  $p^2q$ -type.

For the sake of completeness, we formally define the notion of trapdoor one-way function, resp. permutation.

**DEFINITION 4.1 (COLLECTION OF TRAPDOOR ONE-WAY FUNCTIONS)** *Let  $I$  be a set of indices such that for each  $i \in I$  the sets  $D_i$  and  $\tilde{D}_i$  are finite. Let  $\mathcal{F} = \{f_i \mid f_i : D_i \rightarrow \tilde{D}_i\}_{i \in I}$  be a family of functions. Then  $\mathcal{F}$  is said to be a collection of trapdoor one-way functions if*

1. *There exists a polynomial  $p$  and a probabilistic polynomial time key generator  $\text{KeyGen}$  such that  $\text{KeyGen}$  on input  $1^k$  (the security parameter) outputs a pair  $(i, t_i)$  where  $i \in \{0, 1\}^k \cap I$ ,  $|t_i|_2 < p(k)$ . The data  $t_i$  is denoted the trapdoor information of  $f_i$ .*
2. *The domains  $D_i$  are samplable: There exists a probabilistic polynomial time sampling algorithm  $\mathcal{S}$  that on input  $i \in I$  outputs  $x \in D_i$  uniformly chosen at random.*
3. *The elements of  $\mathcal{F}$  are easy to evaluate: There exists a deterministic polynomial time evaluator  $\text{Eval}$  that on input  $i \in I$ ,  $x \in D_i$  outputs  $f_i(x)$ .*
4. *Inverting the elements of  $\mathcal{F}$  is easy if the trapdoor information is known: There exists a deterministic polynomial time inverter  $\text{Inv}$  such that for all  $x \in D_i$  we have  $f_i(\text{Inv}(t_i, f_i(x))) = f_i(x)$ .*
5. *Inverting the elements of  $\mathcal{F}$  is hard if the trapdoor information is unknown: For every probabilistic polynomial time algorithm  $\mathcal{A}_I$  the following is negligible in  $k$ :*

$$\Pr[(i, t_i) \leftarrow \text{KeyGen}(1^k); x \leftarrow D_i; \tilde{x} \leftarrow \mathcal{A}_I(f_i(x)) : f_i(\tilde{x}) = f_i(x)].$$

Trapdoor one-way permutations are bijective trapdoor one-way functionstrapdoor one-way permutation.

**DEFINITION 4.2 (COLLECTION OF TRAPDOOR ONE-WAY PERMUTATIONS)** *Let  $I$  be a set of indices such that for each  $i \in I$  the sets  $D_i$  and  $\tilde{D}_i$  are finite and of the same order. Let  $\mathcal{F} = \{f_i \mid f_i : D_i \rightarrow \tilde{D}_i\}_{i \in I}$  be a family of bijections. Then  $\mathcal{F}$  is said to be a collection of trapdoor one-way permutations if  $\mathcal{F}$  is a collection of trapdoor one-way functions as specified in Definition 4.1 above.*

- REMARK 4.1
1. When speaking of trapdoor one-way functions, resp. permutations we may omit the phrase “one-way” for the reader’s convenience.
  2. In contrast to strictly mathematical parlance, we do not require that permutations are maps onto itself.
  3. In case of trapdoor permutations the value computed by the inverter must equal the original pre-image  $x$  in properties 4. and 5. of Definition 4.1.

### 4.2.1 The Proposed Trapdoor One-way Function

Throughout this section, let  $p, q$  be primes with  $p \nmid q - 1$  and  $q \nmid p - 1$ . We define  $n = p^2q$ . All of our constructions are based on the following group homomorphism:

DEFINITION 4.3 (THE HOMOMORPHISM  $h$ ) *Let  $p, q$  be primes with  $p \nmid q - 1, q \nmid p - 1$  and  $n = p^2q$ . Then we define:*

$$\begin{aligned} h : \mathbb{Z}_n^\times &\longrightarrow \mathbb{Z}_n^\times \\ x &\mapsto x^n \bmod n \end{aligned}$$

The reason why we do not use standard RSA moduli is the observation that in  $\mathbb{Z}_n^\times$  with  $n = p^2q$  there are elements of order  $p$ :

LEMMA 4.1 *Let  $p, q$  be primes with  $p \nmid q - 1$  and  $n = p^2q$ . Define the set  $\mathcal{S}$  as*

$$\mathcal{S} := \{x \in \mathbb{Z}_n^\times \mid x = 1 + kpq \text{ for an integer } k, 0 < k < p\}.$$

*Then  $\mathcal{S}$  consists of exactly the elements of multiplicative order  $p$  in  $\mathbb{Z}_n^\times$ .*

PROOF Let  $x$  be an element of multiplicative order  $p$  in  $\mathbb{Z}_n^\times$ . Then we have

$$\begin{aligned} x^p = 1 \bmod n &\Rightarrow (x^p = 1 \bmod p \wedge x^p = 1 \bmod q) \\ &\Rightarrow (x = 1 \bmod p \wedge x = 1 \bmod q). \end{aligned}$$

Hence,  $pq \mid x - 1$ , and we conclude  $x \in \mathcal{S}$ .

On the other hand, we have  $1 \notin \mathcal{S}$  and from the binomial expansion formula it is obvious that for all  $x \in \mathcal{S}$  the equality  $x^p = 1 \bmod n$  must hold. Thus, the assertion follows.  $\square$

From Lemma 4.1, we can easily deduce that each element of order  $p$  in  $\mathbb{Z}_n^\times$  reveals the factorization of  $n$ . On this fact we will base the one-wayness of our proposed trapdoor functions. Next, we analyze the relationship between the homomorphism  $h$  and the set  $\mathcal{S}$ :

LEMMA 4.2 *Let  $h$  and  $\mathcal{S}$  be defined as above. Then we have*

$$\ker(h) = \{1\} \cup \mathcal{S}.$$

PROOF Note that as  $p$  is the only non-trivial common factor of  $n$  and  $\varphi(n) = p(p-1)(q-1)$ , we must have

$$x^n = 1 \bmod n \iff x = 1 \vee \text{ord}_n(x) = p.$$

Hence, the kernel of  $h$  consists of 1 and exactly the elements of multiplicative order  $p$  in  $\mathbb{Z}_n^\times$ , i. e., the elements of  $\mathcal{S}$  as defined in Lemma 4.1.  $\square$

As therefore the magnitude of the kernel of  $h$  is exactly  $p$ , we obtain

**COROLLARY 4.1** *The homomorphism  $h$  as defined above is  $p$ -to-1.*

Next, we will prove that  $h$  is collision-resistant, as a collision leads to a non-trivial element of  $\ker(h)$ .

**THEOREM 4.1** *For  $x, y \in \mathbb{Z}_n^\times$  we have*

$$x^n = y^n \pmod n \iff h(x) = h(y) \iff x = y \pmod{pq}.$$

**PROOF** “ $\Leftarrow$ ”: Let  $y = x + kpq$  for  $k \in \mathbb{Z}$ . Then,  $(x + kpq)^n = x^n + nx^{n-1}kpq = x^n \pmod n$ .

“ $\Rightarrow$ ”:  $x^n = y^n \pmod n$  leads to  $xy^{-1} \in \ker(h)$ , consequently  $xy^{-1} = 1 \pmod{pq}$  using Lemma 4.1 and Lemma 4.2. □

Thus, we can factor  $n$  if we can find collisions of  $h$ . Before stating this result explicitly, we formally define the  $p^2q$  factorization assumption:

**DEFINITION 4.4** ( $p^2q$  FACTORIZATION ASSUMPTION) *Define  $I(k) = \{n = p^2q \mid p, q \in \mathcal{PRIMES}(k), p \nmid q-1, q \nmid p-1, p \neq q\}$ . Then the  $p^2q$  Factorization Assumption states that for any PPA  $\mathcal{A}$  the following quantity (called  $\mathcal{A}$ 's advantage) is negligible in  $k$ :*

$$\Pr[n \leftarrow I(k) : \mathcal{A}(n) = (p, q)].$$

In Section 4.2.2, we analyze the reliability of the  $p^2q$  Factorization Assumption.

**COROLLARY 4.2** *If the  $p^2q$  Factorization Assumption holds, then the homomorphism  $h$  is collision-resistant.*

**PROOF** Assume that  $\mathcal{A}$  is a polynomial time algorithm that on input  $n$  determines  $x, y \in \mathbb{Z}_n^\times$  with  $x \neq y$  and  $h(x) = h(y)$ . From Theorem 4.1, we conclude  $\gcd(x - y, n) = pq$ , which completely reveals the factorization of  $n$ . □

In the following, we introduce two restrictions of  $h$  that turn out to be trapdoor permutations. For this purpose, we formally define the set of  $n$ -th residues modulo  $n$ .

**DEFINITION 4.5** (N-R( $n$ )) *Let  $\text{N-R}(n) = \{x \in \mathbb{Z}_n^\times \mid x = y^n \pmod n \text{ for a } y \in \mathbb{Z}_n^\times\} = \text{im}(h)$  denote the set of the  $n$ -th residues modulo  $n$ .*

$\text{N-R}(n)$  is a subgroup of  $\mathbb{Z}_n^\times$  of order  $(p-1)(q-1)$  (as there are exactly  $\varphi(pq) = (p-1)(q-1)$  pairwise different  $n$ -th residues modulo  $n$ , namely the elements  $\{x^n \pmod n \mid x \in \mathbb{Z}_{pq}^\times\}$ ).

Now we can state the main results of this section:

**THEOREM 4.2** *1. Let  $I = \{n = p^2q \mid p, q \in \mathcal{PRIMES}, p \nmid q-1, q \nmid p-1, p \neq q\}$  be a set of indices. The family  $\mathcal{F} = \{h^{(n)}\}_{n \in I}$  is a collection of trapdoor one-way functions, where  $h^{(n)}$  is defined as*

$$\begin{aligned} h^{(n)} : \mathbb{Z}_n^\times &\longrightarrow \text{N-R}(n) \\ x &\mapsto x^n \pmod n. \end{aligned}$$

2. Let  $I$  be defined as above. The family  $\mathcal{F}_{\text{N-R}} = \{h_{\text{N-R}}^{(n)}\}_{n \in I}$  is a collection of trapdoor one-way permutations, where  $h_{\text{N-R}}^{(n)}$  is defined as

$$\begin{aligned} h_{\text{N-R}}^{(n)} : \text{N-R}(n) &\longrightarrow \text{N-R}(n) \\ x &\mapsto x^n \bmod n. \end{aligned}$$

3. Let  $I$  be defined as above. The family  $\mathcal{F}_{pq} = \{h_{pq}^{(n)}\}_{n \in I}$  is a collection of trapdoor one-way permutations, where  $h_{pq}^{(n)}$  is defined as

$$\begin{aligned} h_{pq}^{(n)} : \mathbb{Z}_{pq}^\times &\longrightarrow \text{N-R}(n) \\ x &\mapsto x^n \bmod n. \end{aligned}$$

In all cases, the trapdoor is the factorization of  $n$  and the one-wayness is based on the  $p^2q$  Factorization Assumption. We omit the superscript  $(n)$  whenever it is clear from the context.

**PROOF** 1. Properties 1. to 3. of Definition 4.1 are obviously fulfilled. Define  $d = n^{-1} \bmod \varphi(pq)$  (note that  $\gcd(n, \varphi(pq)) = 1$ ). Then it is easy to see that  $(h(x))^d = x \bmod pq$  holds for all  $x \in \mathbb{Z}_n^\times$ . Thus,  $d$  (resp. the factorization of  $n$ ) can be utilized as a trapdoor to invert  $h$ , which shows property 4. Note that we always obtain the minimum of all pre-images (from Theorem 4.1, we conclude that there is exactly one pre-image smaller than  $pq$ ). The one-wayness (property 5.) is a consequence of Corollary 4.2: To factor  $n$  with access to an oracle that inverts  $h$ , we choose an element  $x \in \mathbb{Z}_n^\times$  at random and query the oracle on  $h(x) = x^n \bmod n$ . With probability  $1 - 1/p$ , the oracle will answer a pre-image  $\tilde{x} \neq x \bmod n$ . Thus,  $x$  and  $\tilde{x}$  collide under  $h$ , and from Corollary 4.2 we deduce that  $\gcd(x - \tilde{x}, n) = pq$  reveals the factorization of  $n$ .

2. We first show that the  $h_{\text{N-R}}$  are indeed permutations. Define  $d$  as above. Let  $x$  be an element of  $\text{N-R}(n)$ , *i. e.*,  $x = y^n \bmod n$  for an appropriate  $y \in \mathbb{Z}_n^\times$ . Then we have  $(x^n)^d = y^{n^2d} = x \bmod n$ , because of  $n^2d = n \bmod \varphi(n)$  (equality holds modulo  $p$  and modulo  $\varphi(pq)$ ). Thus,  $x \mapsto x^n \bmod n$  is a permutation of  $\text{N-R}(n)$ . The remaining properties can be shown along the lines of the preceding proof.
3. The considerations in 1. particularly imply that  $(h_{pq}(x))^d = x \bmod pq$  holds for all  $x \in \mathbb{Z}_{pq}^\times$ . Moreover, we have  $\#\mathbb{Z}_{pq}^\times = (p-1)(q-1) = \#\text{N-R}(n)$ . Thus,  $h_{pq}$  is a bijection. Again, the remaining properties can be shown along the lines of the proof given in 1.

**REMARK 4.2** The fact that modular exponentiation with  $n = p^2q$  can be inverted uniquely modulo  $pq$  has been implicitly exploited in [Pai99b], where Pascal Paillier introduced a trapdoor permutation based on the Okamoto-Uchiyama trapdoor mechanism. However, the results and the proof techniques used in [Pai99b] are substantial different from our proposal. In particular, Paillier does not utilize the one-wayness of raising to the  $n$ -th power, which is a central tool in our constructions.

### 4.2.2 The Hardness of the $p^2q$ Factoring Problem

Recently, the use of  $p^2q$  type moduli (resp. more general  $p^kq$ ) attracted much attention in cryptography. For example, the modulus  $p^2q$  is used in the famous family of EPOC cryptosystems (based on Okamoto-Uchiyama homomorphic encryption) [FKM<sup>+</sup>00, OU98, OP00] and in the signature scheme ESIGN [FOM91], whereas moduli  $p^kq$  can be utilized to enhance the decryption speed in RSA-type encryption schemes [Tak98, Tak04].

Numerous researchers tried to exploit the special form of those integers to find faster factorization methods [AM94, PO96, BDHG99]. But unless the exponent  $k$  in  $p^kq$  is not too large, the most efficient methods for factoring  $n = p^kq$  are still Lenstra's elliptic curve method (ECM) [Len87], its improvements [PO96], and the general number field sieve (GNFS) [LL93]. More precisely, if the size of the smallest prime factor of  $n$  exceeds some bound, the GNFS is the method of choice<sup>3</sup>. Consequently, if  $n$  is sufficiently large (*i. e.*, exceeding 1000 bits), the special form  $n = p^2q$  causes no problem, because in contrast to ECM the runtime of the GNFS only depends on the size of  $n$ , not on the size of its smallest prime factor. Concluding, although it is not known if factoring  $n = p^2q$  is more tractable than factoring  $n = pq$  or not, the  $p^2q$  Factorization Assumption is well-investigated and therefore can be regarded as fairly weak.

### 4.2.3 Comparison

We want to point out the similarities among exponentiation modulo  $n = p^2q$  and Rabin-type modular squaring. In both cases, we have a group homomorphism with a non-trivial kernel (of size four in case of Rabin and of size  $p$  in our case). Moreover, one-wayness holds because each non-trivial kernel element reveals the factorization of the modulus. Obviously, the Blum-Williams permutation on quadratic residues corresponds to our permutation  $h_{N-R}$  on  $n$ -th residues. In the case of modular squaring, however, to the best of our knowledge there is no analogue to the bijection  $h_{pq}$ . The latter is interesting for practical applications, as no preprocessing into the set of  $n$ -th residues is necessary. In particular,  $h_{pq}$  can be used to encrypt *arbitrary* strings like keys. We provide an application in Section 4.4. Further advantages of our proposal are due to the fact that the magnitude of the kernel is larger. To confirm this statement, we will construct fail-stop signatures in Section 4.5 and trapdoor commitments in Section 4.6 from the homomorphism  $h$ . In summary, our proposal suffers from one obvious disadvantage (efficiency), but it also features lots of less conspicuous advantages compared to Rabin-type modular squaring.

To emphasize the analogy to Rabin-type modular squaring even more, we assume that without knowledge of the factors of  $n$  distinguishing N-R( $n$ ) from  $\mathbb{Z}_n^\times$  is hard (*cf.* the well-known quadratic residuosity assumption).

**DEFINITION 4.6 (DECISIONAL COMPOSITE RESIDUOSITY ASSUMPTION)** *Define  $I(k) = \{n = p^2q \mid p, q \in \mathcal{PRIMES}(k), p \nmid q - 1, q \nmid p - 1, p \neq q\}$ . Then the Decisional Composite Residuosity Assumption (DCRA) states that for any polynomial time distinguisher  $\mathcal{A}$  the following quantity (called  $\mathcal{A}$ 's advantage) is negligible in  $k$ :*

$$|\Pr[n \leftarrow I(k); x \leftarrow \mathbb{Z}_n^\times : \mathcal{A}(x) = 1] - \Pr[n \leftarrow I(k); x \leftarrow \mathbb{Z}_n^\times : \mathcal{A}(x^n \bmod n) = 1]|$$

<sup>3</sup>The currently largest factor found with ECM consists of 66 decimal digits (April 2005), whilst the current GNFS record is the factorization of a 200 decimal digits RSA modulus (May 2005).

Given  $p$  and  $q$ , however, deciding  $n$ -th residuosity is efficient.

**THEOREM 4.3** *For all  $x \in \mathbb{Z}_n^\times, x > 1$  we have*

$$x \in \text{N-R}(n) \iff x^{p-1} = 1 \pmod{p^2}.$$

**PROOF** First we show an auxiliary proposition:

$$x \in \text{N-R}(n) \iff x^{(p-1)(q-1)} = 1 \pmod{n}.$$

From  $p \nmid q-1, q \nmid p-1$  we deduce  $\gcd((p-1)(q-1), n) = 1$ . Hence there exists  $z \in \mathbb{Z}$  with  $z(p-1)(q-1) = 1 \pmod{n}$ , leading to  $-z(p-1)(q-1) + 1 = kn$  for a suitable  $k \in \mathbb{Z}$ . Thus, we have

$$\begin{aligned} x^{(p-1)(q-1)} = 1 \pmod{n} &\Rightarrow x^{-z(p-1)(q-1)+1} = x \pmod{n} \\ &\Rightarrow (x^k)^n = x \pmod{n}. \end{aligned}$$

This finishes the proof of the auxiliary proposition, as the opposite direction is straightforward.

Therefore, we have the following for  $x > 1$ :

$$\begin{aligned} x \in \text{N-R}(n) &\iff x^{(p-1)(q-1)} = 1 \pmod{n} \\ &\iff x^{(p-1)(q-1)} = 1 \pmod{p^2} \text{ and } x^{(p-1)(q-1)} = 1 \pmod{q} \end{aligned} \quad (4.1)$$

$$\iff x^{(p-1)(q-1)} = 1 \pmod{p^2} \quad (4.2)$$

$$\iff x^{(p-1)} = 1 \pmod{p^2}. \quad (4.3)$$

Note that (4.2)  $\Rightarrow$  (4.1) holds because  $x^{(p-1)(q-1)} = 1 \pmod{q}$  is true for all  $x \in \mathbb{Z}_n^\times$ . From  $\gcd(q-1, \varphi(p^2) = 1)$  we deduce (4.2)  $\Rightarrow$  (4.3).  $\square$

Table 4.1 summarizes the above considerations.

$\mathbb{Z}_n^\times \longrightarrow \text{N-R}(n) \text{ for } n = p^2q$ $x \mapsto x^n \pmod{n}$	$\mathbb{Z}_n^\times \longrightarrow \text{QR}(n) \text{ for } n = pq$ $x \mapsto x^2 \pmod{n}$
group homomorphism	
$p$ -to-1	$4$ -to-1
non-trivial kernel element reveals factorization of $n$	
restriction to $\text{N-R}(n)$ is permutation	restriction to $\text{QR}(n)$ is permutation
restriction to $\mathbb{Z}_{pq}^\times$ is permutation	no analogue known
hard to distinguish $\text{N-R}(n)$ and $\mathbb{Z}_n^\times$	hard to distinguish $\text{QR}(n)$ and $\mathbb{Z}_n^\times$
above distinction is easy of factors of $n$ are known	

Table 4.1: Comparison between proposed trapdoor function and Rabin-type modular squaring

## 4.3 Application to Homomorphic Encryption

Homomorphic encryption schemes provide the possibility of computing the ciphertext of an appropriate combination of two messages from the ciphertexts of these messages only. Here, combination usually refers to a group operation such as modular addition, modular multiplication, or XOR. Thus, homomorphic encryption enables a third party to operate on the underlying messages without decrypting the ciphertexts, in particular without knowledge of the secret key. Although this property implies malleability and therefore a homomorphic encryption scheme can never meet the strongest security notion IND-CCA2, there are nevertheless several applications for homomorphic encryption as building blocks for cryptographic protocols like electronic voting, private information retrieval, and threshold cryptography.

### Previous Work

In their seminal paper from 1984, Goldwasser and Micali introduced the notion of semantic security and presented the first cryptosystem meeting this requirements [GM84]. Their proposed probabilistic cryptosystem is additively homomorphic but suffers from a very limited bandwidth (the encryption is performed bit-wise). Over the intervening years this scheme has been improved several times, where the most notable ameliorations came from Benaloh-Fischer [CF85] and Naccache-Stern [NS98]. However, the actual breakthrough in the field of semantically secure additive homomorphic encryption has been achieved by Okamoto-Uchiyama and Paillier with a different approach. Namely, their idea was to change the group structure from  $\mathbb{Z}_n^\times$  with a RSA modulus  $n$  to  $\mathbb{Z}_n^\times$  with  $n = p^2q$  (Okamoto-Uchiyama [OU98]), resp. to  $\mathbb{Z}_{n^2}^\times$  (Paillier [Pai99a]). Both works gained recognition not only for presenting practical solutions to homomorphic encryption, but also for pointing out the rich mathematical structure of the groups  $\mathbb{Z}_n^\times$  with  $n = p^2q$ , resp.  $\mathbb{Z}_{n^2}^\times$ ,  $n = pq$ . Whilst the assumptions on which IND-CPA security relies seems to be comparable for both schemes (*p-subgroup assumption* versus *decisional composite residuosity assumption*), this is not the case for one-wayness: Okamoto-Uchiyama's cryptosystem can be proved one-way if factoring integers  $p^2q$  is hard, but for Paillier's scheme, no security proof based on a standard intractability assumption has been observed yet<sup>4</sup>. In the following, several variants of Paillier's original scheme have been described, *e. g.*, RSA-Paillier [CGHGN01], that significantly reduces the encryption costs and that is one-way under the standard RSA-assumption as shown in Section 3.2.2. However, the one-way reduction is not tight and the scheme is not homomorphic any more.

### Our Contributions

Our contribution in this section is the development of a factorization-based variant of Paillier's homomorphic encryption scheme. Our concept is to study Paillier's original encryption function in a different group, *i. e.*, instead of  $\mathbb{Z}_{n^2}^\times$  with an RSA modulus  $n$  we consider the group  $\mathbb{Z}_n^\times$  with the Okamoto-Uchiyama modulus  $n = p^2q$ . Based on the analysis of the trapdoor one-way function introduced in Section 4.2.1, we are able to show that the proposed cryptosystem is one-way under the  $p^2q$  Factorization Assumption. Moreover,

<sup>4</sup>In [Pai99a], Paillier based the one-wayness of his scheme on the *composite residuosity assumption*, but this assumption is merely a paraphrase of the designated one-wayness property.

the new scheme inherits all the nice properties of Paillier's original one, such as semantic security, additively homomorphic property and efficiency. Unfortunately, the new scheme inherits the most serious drawback of Okamoto-Uchiyama's cryptosystem, too, namely it is vulnerable to a simple chosen ciphertext attack of lunchtime-type (in general, this seems to be the flip-side of the coin regarding factorization-based one-wayness, see *e. g.*, textbook Rabin). This problem has to be dealt with on the protocol level. Paillier's original scheme is neither provably secure against lunchtime attacks, nor there are successful attacks known today.

### 4.3.1 Basic Notions and Definitions for Homomorphic Encryption

The most important notions regarding public key encryption have already been presented in Section 2.2. Therefore, we only define the special case of homomorphic encryption and present an example.

**DEFINITION 4.7 (HOMOMORPHIC ENCRYPTION)** *The probabilistic public key encryption scheme  $(\text{KeyGen}, \mathcal{E}, \mathcal{D})$  is said to be*

**homomorphic**, if the message space is a group  $(G, \circ_G)$ , the ciphertext space is a group  $(H, \circ_H)$  and for any  $m_1, m_2 \in G$  we have

$$\mathcal{E}_{\text{pk}}(m_1 \circ_G m_2) = \mathcal{E}_{\text{pk}}(m_1) \circ_H \mathcal{E}_{\text{pk}}(m_2).$$

**entirely homomorphic**, if the space of randomness used for encryption and the message space form a group  $(G, \circ_G)$ , the ciphertext space is a group  $(H, \circ_H)$  and for any  $(m_1, r_1), (m_2, r_2) \in G$  we have

$$\mathcal{E}_{\text{pk}}((m_1, r_1) \circ_G (m_2, r_2)) = \mathcal{E}_{\text{pk}}(m_1; r_1) \circ_H \mathcal{E}_{\text{pk}}(m_2; r_2).$$

**REMARK 4.3** The definition of *entirely homomorphic* is as general as possible. In common examples, the message space and the space of randomness can also be described by two separate groups. Our proposed scheme, however, meets the above specified requirement. For most applications, homomorphic encryption is sufficient.

In Section 2.2.3, we have already observed that Paillier's encryption scheme is entirely homomorphic with  $G = \mathbb{Z}_n \times \mathbb{Z}_n^\times$  and  $(m_1, r_1) \circ_G (m_2, r_2) = (m_1 + m_2 \bmod n, r_1 r_2 \bmod n^2)$  due to  $g^{m_1 + m_2 \bmod n} (r_1 r_2)^n = (g^{m_1} r_1) (g^{m_2} r_2^n) \bmod n^2$  (recall that  $n$  divides the order of  $g$  modulo  $n^2$ ).

We additionally provide a second example that is important for the rest of this section.

**EXAMPLE 4.1 (OKAMOTO-UCHIYAMA ENCRYPTION SCHEME)** We briefly sketch the Okamoto-Uchiyama encryption scheme from Eurocrypt 1998 [OU98]. Let  $n$  be of the shape  $n = p^2 q$  for two large primes  $p, q$ . Consider the Sylow group  $\Gamma_p = \{x \in \mathbb{Z}_{p^2} \mid x \equiv 1 \pmod{p}\}$  of  $\mathbb{Z}_{p^2}^\times$ . The crucial observation is that the  $L$ -function defined on  $\Gamma_p$  as  $L_p(x) = (x - 1)/p$  provides additive homomorphic properties. The encryption scheme is described as follows:

**KeyGen** : On the input  $1^k$  (security parameter) choose  $p, q \in \text{PRIMES}(k), p \neq q$  and define  $n = p^2 q$ . Furthermore choose  $h \in \text{N-R}(n)$ , and  $g \in \mathbb{Z}_n^\times$  with  $p \mid \text{ord}_{p^2}(g)$ . The public key equals  $(n, h, g, k)$ , whilst  $p$  is the secret key. Moreover we have  $\mathcal{M} = \{0, 1\}^{k-1}, \mathcal{C} = \mathbb{Z}_n$ .

$\mathcal{E}$  : Choose a random value  $r \in \mathbb{Z}_n$ . For  $m \in \mathcal{M}$ , the ciphertext is computed as  $c = g^m h^r \bmod n$ .

$\mathcal{D}$  : A ciphertext  $c \in \mathcal{C}$  is decrypted in the following way:  $c' = c^{p-1} \bmod p^2$ ,  $g_p = g^{p-1} \bmod p^2$ ,  $m = L(c')L(g_p)^{-1} \bmod p$ .

The correctness is deduced from the additive homomorphic properties of the  $L$ -function because we have  $c' = g_p^m \bmod p^2$  and  $g_p \in \Gamma_p$ . To enhance the decryption cost, the value  $g_p$  is usually precomputed and stored in the secret key. In [OU98] it is shown that breaking the one-wayness of this scheme is as hard as factoring the modulus. Moreover, the scheme is IND-CPA under the  $p$ -subgroup assumption.

### 4.3.2 The Proposed Homomorphic Encryption Scheme

Our new encryption scheme is based on the following theorem:

**THEOREM 4.4** *Let  $n = p^2q$  for primes  $p, q > 3$  with  $p \nmid q - 1, q \nmid p - 1$ . The map  $f : \mathbb{Z}_n^\times \times \mathbb{Z}_n \longrightarrow \mathbb{Z}_{n^2}^\times$ ,  $(r, m) \mapsto r^n(1 + mn) \bmod n^2$  has the following properties:*

1.  $f$  is well-defined, i. e., if  $r = r' \bmod n$  and  $m = m' \bmod n$  holds, then it follows that  $r^n(1 + mn) = r'^n(1 + m'n) \bmod n^2$  is true.
2.  $f$  is homomorphic in  $r$  and  $m$ , i. e.,  $f(r_1 r_2, m_1 + m_2) = f(r_1, m_1) f(r_2, m_2)$ .
3.  $f(r, m) = f(r + ipq, m - ir^{-1}pq)$  for  $i \in \mathbb{Z}$ , hence  $f$  is  $p$ -to-one.
4.  $\text{im}(f) = \{x = x_0 + nx_1 \in \mathbb{Z}_{n^2}^\times \mid x_0 \in \text{N-R}(n), x_1 \in \mathbb{Z}_n\}$ .
5. The restrictions  $f_m = f|_{\mathbb{Z}_n^\times \times \mathbb{Z}_{pq}}$  and  $f_r = f|_{\mathbb{Z}_{pq}^\times \times \mathbb{Z}_n}$  are one-to-one.
6.  $f_r : \mathbb{Z}_{pq}^\times \times \mathbb{Z}_n \longrightarrow \mathbb{Z}_{n^2}^\times$  is a group homomorphism with respect to the group operation  $\circ_r$  on  $\mathbb{Z}_{pq}^\times \times \mathbb{Z}_n$ :

$$(r_1, m_1) \circ_r (r_2, m_2) = \left( \underbrace{r_1 r_2 \bmod pq}_{=: r_{pq}}, m_1 + m_2 + lr_{pq}^{-1}pq \bmod n \right),$$

where  $0 \leq l < p$  is defined via  $r_1 r_2 = r_{pq} + lpq \bmod n$ .

7.  $f_m : \mathbb{Z}_n^\times \times \mathbb{Z}_{pq} \longrightarrow \mathbb{Z}_{n^2}^\times$  is a group homomorphism with respect to the group operation  $\circ_m$  on  $\mathbb{Z}_n^\times \times \mathbb{Z}_{pq}$ :

$$(r_1, m_1) \circ_m (r_2, m_2) = \left( r_1 r_2 - lpq \bmod n, \underbrace{m_1 + m_2 \bmod pq}_{=: m_{pq}} \right),$$

where  $0 \leq l < p$  is defined via  $m_1 + m_2 = m_{pq} - (r_1 r_2)^{-1}lpq \bmod n$ .

**PROOF** 1. Straightforward (use  $(r + in)^n = \sum_{j=0}^n \binom{n}{j} r^{n-j} (in)^j = r^n \bmod n^2$ ).

2. Straightforward computation.

3. We have

$$\begin{aligned}
f(r + ipq, m - ir^{-1}pq) &= (r + ipq)^n(1 + (m - ir^{-1}pq)n) \\
&= \left( \sum_{j=0}^3 \binom{n}{j} r^{n-j} (ipq)^j \right) (1 + (m - ir^{-1}pq)n) \\
&= \left( r^n + nr^{n-1}ipq + \frac{n(n-1)}{2} r^{n-2} (ipq)^2 + \frac{n(n-1)(n-2)}{6} r^{n-3} (ipq)^3 \right) \\
&\quad (1 + (m - ir^{-1}pq)n) \\
&= (r^n + nr^{n-1}ipq)(1 + (m - ir^{-1}pq)n) \\
&= r^n + n(r^{n-1}ipq + r^n m - r^n ir^{-1}pq) = r^n(1 + mn) = f(r, m) \pmod{n^2}
\end{aligned}$$

Note that the second step is true, because  $n^2 = p^4 q^2$  is an integer divisor of  $(ipq)^j$  for  $j > 3$ . The fourth step holds because  $n$  divides  $n(n-1)/2$  as well as  $n(n-1)(n-2)/6$  ( $n$  is odd and either  $n-1$  or  $n-2$  is a multiple of 3).

4. - 7. are more or less immediate consequences of 3. and 2. □

We can now define our proposed encryption scheme as a variant of Paillier's scheme described in Example 2.2.

**KeyGen** : Let  $k$  be a security parameter. Choose  $p, q \in \mathcal{PRIMES}(k)$  such that  $p \nmid q-1, q \nmid p-1, p \neq q$  and define  $n = p^2 q$ . Compute  $d = n^{-1} \pmod{(p-1)(q-1)}$  and let  $l$  be chosen such that  $2^l < pq < 2^{l+1}$ .

The public key is  $\text{pk} = (n, l)$ , and the secret key is  $\text{sk} = (d, p, q)$ . Moreover, set  $\mathcal{M} = \{0, 1\}^l, \mathcal{C} = \mathbb{Z}_{n^2}^\times$ .

**$\mathcal{E}$**  : To encrypt a message  $m \in \mathcal{M}$ , choose  $r \in \mathbb{Z}_n^\times$  at random and compute the ciphertext as  $c = r^n(1 + mn) \pmod{n^2}$ .

**$\mathcal{D}$**  : To decrypt a ciphertext  $c \in \mathcal{C}$ , first compute  $r = c^d \pmod{pq}$ . Then  $m$  is recovered as  $m = L_n(r^{-n}c \pmod{n^2}) \pmod{pq}$ , where the  $L$ -function is defined as  $L_n(x) = \frac{x-1}{n}$ .

We can prove the following theorem:

**THEOREM 4.5** 1. *The proposed encryption scheme is correct.*

2. *The proposed encryption scheme is entirely homomorphic, i. e., we have*

$$\mathcal{E}_{\text{pk}}(m_1; r_1) \mathcal{E}_{\text{pk}}(m_2; r_2) = \mathcal{E}_{\text{pk}}((m_1, r_1) \circ (m_2, r_2)),$$

where  $\circ$  is the group operation on  $\mathbb{Z}_n^\times \times \mathbb{Z}_{pq}$  as defined in Theorem 4.4(7).

3. *The proposed encryption scheme is OW-CPA under the  $p^2q$  Factorization Assumption.*

PROOF 1. Let  $c = r^n(1 + mn) \bmod n^2$  be a ciphertext for  $m \in \mathbb{Z}_{pq}$  and  $r \in \mathbb{Z}_n^\times$ . Define  $\tilde{r} = c^d \bmod pq$ . From Theorem 4.4(5) we know that  $f$  is one-way if the space of randomness or the space of messages is restricted modulo  $pq$ . Therefore, we obtain  $p$  different messages modulo  $n$  if we compute  $L_n(r_i^{-n}c \bmod n^2)$  for  $r_i := \tilde{r} + ipq \bmod n^2, i = 0, \dots, p-1$ . However, Theorem 4.4(3) tells us that all these messages are congruent modulo  $pq$ . Hence,  $L_n(\tilde{r}^{-n}c \bmod n^2)$  uniquely recovers the original  $m < pq$ .

2. See Theorem 4.4(7).

3. Assume that there is a PPA  $\mathcal{A}$  winning GAME.OW as described in Definition 2.3 with non-negligible probability  $\varepsilon$ . We show how to factor  $n$  by playing the following modified game with  $\mathcal{A}$ : First, we choose  $m' \in \mathbb{Z}_n$  and  $r \in \mathbb{Z}_n^\times$  at random and build the fake ciphertext  $c' = r^n(1 + m'n) \bmod n^2$ . In step 4,  $\mathcal{A}$  is given the public key  $n, l$  and the challenge  $c'$ . We distinguish two cases:

**Case 1:** We have  $|m' \bmod pq|_2 \leq l$ . From Theorem 4.4(3), we conclude that in this case the distribution of  $c'$  is exactly the same as the distribution of the original ciphertexts. Moreover, we deduce that  $c' = \mathcal{E}_{\text{pk}}(m' \bmod pq; r')$  holds for an appropriate  $r' \in \mathbb{Z}_n^\times$  with  $r = r' \bmod pq$ . Thus,  $\mathcal{A}$  will return  $m'_{pq} := m' \bmod pq$  with probability  $\varepsilon$ . From the choice of  $m'$ , we observe that  $m' > pq$  holds with probability  $1 - 1/p$ . Thus, we factor  $n$  via  $\gcd(n, m' - m'_{pq}) = pq$ . From the definition of  $l$ , we conclude that this case holds with probability greater than  $1/2$ .

**Case 2:** We have  $|m' \bmod pq|_2 > l$ . In this case,  $c'$  is not an element of the regular ciphertext space and we cannot predict the adversary  $\mathcal{A}$ 's advantage in the modified game.

Therefore, we factor the modulus with advantage at least  $\frac{\varepsilon}{2}(1 - 1/p)$ .

□

REMARK 4.4 Although the group operation that makes the scheme entirely homomorphic is non-standard, we want to point out that for almost all applications the following homomorphic property of the proposed scheme is sufficient:  $\mathcal{E}_{\text{pk}}(m_1; r_1)\mathcal{E}_{\text{pk}}(m_2; r_2) = \mathcal{E}_{\text{pk}}(m_1 + m_2 \bmod pq; r)$  for some  $r \in \mathbb{Z}_n^\times$ .

We now show that the proposed scheme is indistinguishable against passive adversaries under the Decisional Composite Residuosity Assumption (DCRA) introduced in Definition 4.6 lifted to the domain  $\mathbb{Z}_{n^2}^\times$ . This assumption—abbreviated DSCRA for *Decisional Squared Composite Residuosity Assumption* hereafter—states that  $n$ -th residues in  $\mathbb{Z}_{n^2}^\times$  are indistinguishable from non-residues. As each  $x \in \mathbb{Z}_{n^2}^\times$  has a unique  $n$ -adic decomposition  $x = x_0 + nx_1 \bmod n^2, 0 \leq x_0, x_1 < n, x_0 \in \mathbb{Z}_n^\times$ , and because we have  $x^n = (x_0 + nx_1)^n = x_0^n + \binom{n}{1}x_1^n = x_0^n \bmod n^2$ , the following equality holds:  $\{x^n \bmod n^2 | x \in \mathbb{Z}_{n^2}^\times\} = \{x^n \bmod n^2 | x \in \mathbb{Z}_n^\times\}$ . Thus, to obtain the DSCRA from Definition 4.6, we only have to replace  $x \leftarrow \mathbb{Z}_n^\times$  by  $x \leftarrow \mathbb{Z}_{n^2}^\times$  in the left probability, and  $\mathcal{A}(x^n \bmod n) = 1$  by  $\mathcal{A}(x^n \bmod n^2) = 1$  in the right probability. Recall that this assumption only differs from Paillier's original security assumption in the type of the modulus ( $p^2q$  instated of  $pq$ ). As long as factoring integers of  $p^2q$  type is supposed to be as hard as factoring integers of  $pq$  type, there is no reason to believe that the tractability of DSCR problem depends on the type of the modulus.

**THEOREM 4.6** *The proposed scheme is IND-CPA if and only if the Decisional Squared Composite Residuosity Assumption holds.*

**PROOF** To prove that the proposed scheme is IND-CPA under the DSCRA, we construct a distinguisher  $\mathcal{D}$  which breaks DSCRA using the adversary  $\mathcal{A}$  against the indistinguishability of the proposed scheme as a subroutine. Assume that  $\mathcal{A}$  wins GAME.IND as described in Definition 2.4 with non-negligible probability. Let  $x \in \mathbb{Z}_{n^2}^\times$  be an instance of the DSCR problem.  $\mathcal{D}$  plays the following modified game with  $\mathcal{A}$ . In step 2,  $\mathcal{D}$  runs  $\mathcal{A}$  on the public key and obtains a state information  $st$  and two different messages  $m_0, m_1 \in \mathcal{M}$ . Then  $\mathcal{D}$  chooses a bit  $b \in \{0, 1\}$  at random, computes  $c = x(1 + m_b n) \bmod n^2$ , and runs  $\mathcal{A}$  on  $(st, c)$  (step 4).  $\mathcal{D}$  returns 1 (indicating that  $x$  is a  $n$ -th residue modulo  $n^2$ ) if  $\mathcal{A}$ 's answer is  $b$ , otherwise,  $\mathcal{D}$  returns 0. If  $x$  is an  $n$ -th residue modulo  $n^2$ , then  $c$  is a valid cipher of  $m_0$  (distributed as original ciphertexts), and consequently the modified game and the original game are perfectly indistinguishable from  $\mathcal{A}$ 's point of view. Otherwise,  $c$  is a random element of  $\mathbb{Z}_{n^2}^\times$  (independent from everything  $\mathcal{A}$  knows), and  $\mathcal{A}$ 's advantage must equal  $1/2$ . Thus, the overall advantage of  $\mathcal{D}$  equals  $\varepsilon/2$ , where  $\varepsilon$  is the non-negligible advantage of  $\mathcal{A}$  in winning GAME.IND.

To prove the opposite direction, we sketch how a distinguisher  $\mathcal{D}$  of DSCRA with non-negligible advantage  $\varepsilon$  can be used to break the indistinguishability of the proposed scheme. The adversary  $\mathcal{A}$  against IND-CPA play GAME.IND as follows: In step 2,  $\mathcal{A}$  chooses two different messages  $m_0, m_1 \in \mathcal{M}$  and a bit  $b \in \{0, 1\}$  at random. Given the challenge  $c \in \mathbb{Z}_{n^2}^\times$  in step 5,  $\mathcal{A}$  queries  $\mathcal{D}$  on  $c(1 - m_b n) \bmod n^2$  and returns  $b$  if the answer is 1,  $1 - b$  otherwise (note  $(1 + m_b n)^{-1} = (1 - m_b n) \bmod n^2$ ). With a similar argumentation as above we conclude that the advantage of  $\mathcal{A}$  equals  $\varepsilon/2$ .  $\square$

### 4.3.3 Comparison

Summing up, we showed that by replacing the modulus  $n = pq$  with  $n = p^2q$  in Paillier's scheme one obtains an encryption scheme with the following properties:

- The proposed scheme is one-way under a weak standard assumption (namely factoring  $n = p^2q$ ).
- The security reduction is tight.
- The proposed scheme is IND-CPA under the same security assumption as Paillier's original one.
- The proposed scheme is still homomorphic.
- The only drawback of the new scheme is that as the Okamoto-Uchiyama encryption it is vulnerable to a simple chosen ciphertext attack of lunchtime-type.

We recapitulate our results in Table 4.2. As the efficiency is similar in all schemes, we do not compare encryption/decryption costs.

Scheme	Assumption (OW)	Assumption (IND-CPA)	CCA1-secure
Okamoto-Uchiyama	<b>FACT</b>	$p$ -subgroup	NO
Paillier	composite residuosity	DSCR	<b>unknown</b>
Proposed	<b>FACT</b>	DSCR	NO

Table 4.2: Comparison of homomorphic encryption schemes

## 4.4 Application to CCA Secure Hybrid Encryption

Although the concept of public key cryptography (*a. k. a.* asymmetric cryptography) is pretty appealing and has many organizational advantages, secret key cryptography (*a. k. a.* symmetric cryptography) is still much more efficient. Thus, for practical applications the combination of both concepts, *i. e.*, *hybrid encryption* is quite popular.

### Previous Work

Combining symmetric and asymmetric encryption has been widely used in practice, although most constructions accrued from ad-hoc considerations and were not rigorously analyzed. In provably secure cryptography, hybrid encryption first played a role in the context of generic constructions for CCA secure public key encryption [BR94, FO99a, FO99b, OP01]. These techniques exploit Vernam’s one-time pad and random oracles to enhance the security of weakly secure public-key primitives like OW-CPA and IND-CPA secure encryption. In [FO99b], Fujisaki and Okamoto were the first to analyze the possibility of replacing the one-time pad by any symmetric encryption meeting appropriate security requirements. Unfortunately, the random oracle paradigm random oracle model is a crucial requirement for all these conversions.

Beside the technique of applying specific generic constructions to suitable asymmetric and symmetric primitives, a more general solution of hybrid encryption has been introduced by Cramer and Shoup in [CS04]. In this paper, Cramer and Shoup formalize the so-called KEM/DEM framework where KEM is a probabilistic asymmetric *key-encapsulation mechanism*, and DEM is a symmetric encryption scheme (a *data encapsulation mechanism*) used to encrypt messages of arbitrary length with the key given by the KEM. As said before, such combinations of public and secret key schemes have been folklore for years, but Cramer and Shoup for the first time gave a rigorously analyzed formal treatment of this subject. Note that a KEM is not the same as a key agreement protocol: the encapsulated key is designated to be used once only, therefore the DEM is only required to be secure in the one-time scenario. For more details on security definitions and requirements the reader is referred to [CS04]. Roughly speaking, if both the KEM and the DEM part are CCA-secure, then the same holds for the whole KEM/DEM scheme.

At Eurocrypt 2005, Abe *et al.* enhanced Cramer and Shoup’s framework by introducing the notion of a *Tag-KEM*, which is a KEM equipped with a special piece of information, the tag [AGK05]. In their novel framework for hybrid encryption, this tag as part of an CCA-secure Tag-KEM is assigned to protect the non-malleability of the DEM part. Consequently, for the CCA-security of the whole Tag-KEM/DEM hybrid scheme with an CCA-secure Tag-KEM, it is only required that the DEM part is secure against *passive* adversaries. This is an obvious improvement compared to the KEM/DEM framework, but the flip-side of the coin is that proving a Tag-KEM to be CCA-secure is somewhat more

involved than the analogue proof for a “plain” KEM.

## Our Contributions

In [AGK05], the authors provide some generic constructions for Tag-KEMs built from combinations of primitives like KEM, MAC, hash-functions and public key encryption. A generic construction from trapdoor permutations, however, is not given. In the following, we construct a new Tag-KEM based on our proposed trapdoor permutation  $h_{pq}$  and prove its CCA-security in the ROM. These considerations can be easily generalized to arbitrary trapdoor permutations, provided the permutations’ domain can be identified with the set of bit-strings of a certain length (as it is the case for  $h_{pq}$ ). Then we show how this leads to a CCA-secure hybrid encryption scheme in the Tag-KEM/DEM framework. Finally, we compare our novel scheme with the members of the well-known EPOC family [FKM<sup>+</sup>00, OP00]. We choose these schemes as a candidate because they all rely on the Okamoto-Uchiyama trapdoor mechanism introduced in Example 4.1, which like ours is based on the  $p^2q$  Factorization Assumption. However, as EPOC-1 has a worse security reduction than EPOC-2 and a similar performance, we focus on EPOC-2 and EPOC-3.

### 4.4.1 Basic Notions and Definitions for Hybrid Encryption

Informally, hybrid encryption can be defined as a branch of public key encryption schemes that uses symmetric encryption as a black box to enhance efficiency of the entire construction. Symmetric encryption can be defined similarly to Definition 2.1, except that there is only a single key used for both encryption and decryption. In addition, symmetric encryption is deterministic. Instead of providing formal definitions for hybrid encryption, we start with giving some well-known examples to illustrate the concept.

**EXAMPLE 4.2 (REACT)** A common approach for designing CCA secure hybrid encryption is to apply generic conversions to appropriate primitives. We sketch the REACT conversion technique proposed by Okamoto and Pointcheval (for details see [OP01]). It is composed of a symmetric encryption scheme  $(\mathcal{E}^{sym}, \mathcal{D}^{sym})$ , a public key encryption scheme  $(\text{KeyGen}^{asym}, \mathcal{E}^{asym}, \mathcal{D}^{asym})$ , a key derivation function KDF and a hash function  $H$ . The key generation for the hybrid scheme is the same as in  $\text{KeyGen}^{asym}$ . In addition, appropriate system parameters are selected. Let  $rLen$  be the key-length of  $(\mathcal{E}^{sym}, \mathcal{D}^{sym})$ . Encryption and decryption are performed as follows:

$$\begin{array}{l|l}
 \mathcal{E}_{pk}(m) : & \mathcal{D}_{sk}(c) : \\
 r \leftarrow \{0, 1\}^{rLen} & (c_1, c_2, c_3) := c \\
 K := \text{KDF}(r) & r := \mathcal{D}_{sk}^{asym}(c_1) \\
 c_1 \leftarrow \mathcal{E}_{pk}^{asym}(r) & K := \text{KDF}(r) \\
 c_2 := \mathcal{E}_K^{sym}(m) & m := \mathcal{D}_K^{sym}(c_2) \\
 c_3 := H(r, m, c_1, c_2) & \text{if } H(r, m, c_1, c_2) \neq c_3 \text{ return } \perp \text{ and halt} \\
 \text{Return } c = (c_1, c_2, c_3) & \text{Return } m
 \end{array}$$

First, a random string  $r$  is selected and put into the key derivation function to obtain a random session key  $K$ . In addition,  $r$  is encrypted with the asymmetric scheme to  $c_1$ . Then the message  $m$  is encrypted using the symmetric scheme with the secret key  $K$ , the result

is  $c_2$ . Finally, a checksum  $H(r, m, c_1, c_2)$  is added to complete the ciphertext. Decryption is done in the obvious way.

Okamoto and Pointcheval prove the following security statement: If  $(\mathcal{E}^{sym}, \mathcal{D}^{sym})$  is one-time secure and  $(\text{KeyGen}^{asym}, \mathcal{E}^{asym}, \mathcal{D}^{asym})$  is OW-PCA, then the resulting REACT scheme is IND-CCA in the ROM (both of KDF and  $H$  are modeled as random oracles). One-time security for symmetric encryption is defined similar to IND-CPA (in particular, the symmetric key is used once only). PCA denotes plaintext-checking attack. In this model, the adversary has access to an oracle that on input a message  $m$  and a ciphertext  $c$  answers if  $c$  is a possible encryption of  $m$ . Of course, this oracle is only helpful if the encryption is probabilistic, otherwise the adversary can answer the queries herself. Thus, in the deterministic scenario, OW-PCA is equivalent to one-wayness under the weakest attack, *i. e.*, chosen-plaintext-attack (CPA). The benefit of REACT is that the security reduction is tight and that the decryption process is very fast, as only the computation of a single hash-value is necessary to check if the ciphertext is well-formed. In previous conversion techniques like Fujisaki-Okamoto [FO99b, FO99a], a costly re-encryption was necessary to fulfill this purpose.

**EXAMPLE 4.3 (EPOC-3)** The most popular encryption scheme obtained from the REACT conversion is EPOC-3, the outcome of the application to Okamoto-Uchiyama encryption introduced in Example 4.1. It is remarkable that although the one-wayness of the Okamoto-Uchiyama encryption scheme is equivalent to factoring integers  $p^2q$ , the security of the converted scheme is only based on the probably stronger Gap-High-Residuosity Assumption (informally, this assumption states that factoring is infeasible even with the help of an Okamoto-Uchiyama plaintext checking oracle). This is due to the fact that Okamoto-Uchiyama is probabilistic and thus OW-PCA is not equivalent to OW-CPA<sup>5</sup>. We do not give a more detailed description of EPOC-3 here, because it is easily obtained by combining Example 4.2 with Example 4.1.

### The Tag-KEM/DEM framework for hybrid encryption

In the following, we review some formal definitions related to the above described Tag-KEM/DEM framework [AGK05]. We focus on those notions that are necessary for presenting our own results in the subsequent section.

**DEFINITION 4.8 (TAG-KEM)** *A Tag-KEM is defined as a quadruple  $(TKEM.Gen, TKEM.Key, TKEM.Enc, TKEM.Dec)$  of polynomial time algorithms such that the following holds:*

1.  *$TKEM.Gen$  is a probabilistic algorithm which on input  $1^k$  (the security parameter) produces a pair  $(pk, sk)$  of public and secret key. For notational simplicity, we assume that the public key  $pk$  implicitly includes a description of all the relevant spaces and additional tools like hash functions, if applicable.*
2.  *$TKEM.Key$  is a probabilistic algorithm which on input  $pk$  outputs a one-time key  $dk$  for the designated DEM and a key carrier  $\omega$ .*

---

<sup>5</sup>One could ask why the randomization is not removed before applying REACT (this would lead to the enciphering  $c = g^m \bmod n$ , and the same decryption as in the original scheme). But note that in this case, we cannot reduce factoring to the one-wayness as before, because the distributions of  $\{g^m \bmod n \mid m > p\}$  and  $\{g^m \bmod n \mid m < p\}$  are not necessarily the same.

3.  $TKEM.Enc_{pk}$  is a probabilistic algorithm which on input  $(\omega, \tau)$  encrypts the embedded one-time key  $dk$  along with a tag  $\tau$  to the cipher  $\Psi$ .
4.  $TKEM.Dec_{sk}$  is a deterministic algorithm which on input  $(\Psi, \tau)$  decapsulates  $\Psi$  along with the tag  $\tau$  and reconstructs  $dk$ .
5. If  $(pk, sk)$  is a possible outcome of  $TKEM.Gen$  and  $\tau$  is a possible tag, then we have  $TKEM.Dec_{sk}(TKEM.Enc_{pk}(\omega, \tau), \tau) = dk$  for all  $(\omega, dk) \leftarrow TKEM.Key(pk)$ .

CCA-security of a Tag-KEM requires that an adversary with adaptive oracle access to  $TKEM.Dec_{sk}$  has no chance to distinguish whether a given one-time key  $dk$  is encapsulated in a challenge  $(\Psi, \tau)$  or not, even if the tag  $\tau$  is chosen by the adversary herself. As usual, this is defined via an appropriate game:

**DEFINITION 4.9 (CCA SECURITY OF TAG-KEM)** *Let  $\mathcal{K}_D$  be the key space of an appropriate DEM, let  $\mathcal{O}$  denote access to the decapsulation oracle  $TKEM.Dec_{sk}(\cdot, \cdot)$  and to the random oracles, if applicable, and let  $\mathcal{A}_T$  be an adversary against Tag-KEM playing the following game:*

**GAME.TKEM:**

Step 1.  $(pk, sk) \leftarrow TKEM.Gen(1^k)$

Step 2.  $\nu_1 \leftarrow \mathcal{A}_T^{\mathcal{O}}(pk)$

Step 3.  $(\omega, dk_1) \leftarrow TKEM.Key(pk), dk_0 \leftarrow \mathcal{K}_D, b \leftarrow \{0, 1\}$

Step 4.  $(\tau, \nu_2) \leftarrow \mathcal{A}_T^{\mathcal{O}}(\nu_1, dk_b)$

Step 5.  $\Psi \leftarrow TKEM.Enc_{pk}(\omega, \tau)$

Step 6.  $\tilde{b} \leftarrow \mathcal{A}_T^{\mathcal{O}}(\nu_2, \Psi)$

*In Step 6. the adversary is restricted not to query the decapsulation oracle on the challenge  $(\Psi, \tau)$ , but queries  $(\Psi, \tilde{\tau})$  for  $\tau \neq \tilde{\tau}$  are permitted. The values  $\nu_1, \nu_2$  are internal state informations. We define the advantage of the adversary  $\mathcal{A}_T$  as  $\epsilon_{\mathcal{A}_T} = \left| \Pr[\tilde{b} = b] - \frac{1}{2} \right|$  and  $\epsilon$  as  $\max_{\mathcal{A}_T}(\epsilon_{\mathcal{A}_T})$ , where the maximum is taken over all adversaries modeled as polynomial time Turing machines. Tag-KEM is said to be CCA-secure, if  $\epsilon$  is negligible in the security parameter  $k$ .*

#### 4.4.2 The Proposed Tag-KEM

In the following we show how to build a CCA-secure Tag-KEM from the trapdoor permutation  $h_{pq}$  introduced in Section 4.2.1. These considerations can be generalized easily to obtain a generic construction of CCA-secure Tag-KEM from trapdoor permutations, provided the permutations' domain can be identified with the set of bit-strings of a certain length. The latter is the case for our proposed trapdoor permutation and the RSA function, but not for the Blum-Williams, resp. Rabin-Williams permutation.

**TKEM.Gen( $1^k$ ):** Let  $k$  be a security parameter. Choose  $p, q \in \mathcal{PRIMES}(k)$  with  $p \nmid q - 1, q \nmid p - 1, p \neq q$ . Build the product  $n = p^2q$ , compute  $d = n^{-1} \bmod \varphi(pq)$ , and let  $rLen$  be chosen such that  $2^{rLen} < pq < 2^{rLen+1}$ . Select a key derivation function KDF which maps bit-strings into the key-space of the designated DEM and a hash-function  $H$ , which outputs bit-strings of length  $hashLen$ . Return a pair  $(pk, sk)$  of public and secret key, where  $pk = (n, rLen, KDF, H)$  and  $sk = (d, p, q)$ .

**TKEM.Key( $pk$ ):** Choose  $\omega \in \{0, 1\}^{rLen}$  uniformly at random, compute  $dk = KDF(\omega)$  and return  $(\omega, dk)$ .

**TKEM.Enc<sub>pk</sub>( $\omega, \tau$ ):** Given the key carrier  $\omega$  and a tag  $\tau$ , compute  $c_1 = \omega^n \bmod n, c_2 = H(\omega, \tau)$  and return  $\Psi = (c_1, c_2)$ .

**TKEM.Dec<sub>sk</sub>( $\Psi, \tau$ ):** Given the encapsulated key  $\Psi$  and a tag  $\tau$ , parse  $\Psi$  to  $c_1, c_2$  and compute  $r = c_1^d \bmod pq$ . If  $|r|_2 > rLen$  or  $H(r, \tau) \neq c_2$ , then return  $\perp$ , return  $KDF(r)$ , otherwise.

In the first step, a key pair is generated. Then a one-time key  $dk$  for the DEM part is constructed by applying a key derivation function to a random bit string. Note that the role of KDF is not only to format the bit string according to the key space of the designated DEM, but also KDF is required to destroy all algebraic relations between its input and the encapsulated key. In the security proof, both of KDF and  $H$  are modeled as random oracles. In the step  $TKEM.Enc_{pk}$ , the one-time key (which in some sense is embedded in  $\omega$ ) is encrypted together with the tag  $\tau$ . Finally, using  $TKEM.Dec_{sk}$ , the one-time key  $dk$  can be recovered from the encapsulation  $\Psi$  and the tag  $\tau$ .

**REMARK 4.5** In the decapsulation procedure, it is necessary to check if  $r$  indeed meets the length requirements ( $|r|_2 \leq rLen$ ), because otherwise a simple chosen-ciphertext attack can be mounted to obtain the secret factor  $pq$  by binary search [JQY01].

We can prove the following theorem:

**THEOREM 4.7** *If factoring integers of the shape  $n = p^2q$  is hard, then the Tag-KEM defined above is CCA-secure in the random oracle model. More formally: If there exists an adversary  $\mathcal{A}_T$  attacking the proposed Tag-KEM in the random oracle model random oracle model as in Definition 4.9*

- in time  $t$ ,
- with advantage  $\epsilon$ ,
- querying the random oracle representing the key derivation function at most  $q_K$  times,
- querying the random oracle representing the hash function at most  $q_H$  times,
- invoking the decapsulation oracle at most  $q_D$  times,

then there exists an adversary  $\mathcal{A}_{Fact}$  who factors  $n = p^2q$  in time  $t'$  and with advantage  $\epsilon'$ , where

$$t' \leq t + t_{gcd}(qH) + t_{h_{pq}}qD,$$

$$\epsilon' \geq \left( \epsilon - \frac{q_K}{KLen} - \frac{2q_D}{hashLen} - \frac{q_D}{n + hashLen} \right) \left( 1 - \frac{1}{p} \right),$$

where  $t_{\text{gcd}}$  is the time needed to perform a gcd computation with inputs  $\mathcal{O}(n)$ ,  $t_{h_{pq}}$  is the time needed to evaluate  $h_{pq}$ , and  $KLen, hashLen$  are the output lengths of KDF and the hash function  $H$ , respectively.

**PROOF** We prove Theorem 4.7 using a series of games. Let  $\mathcal{A}_T$  be an attacker against the CCA-security of the proposed Tag-KEM and let  $y^*$  be defined as  $y^* = h_{pq}(\omega^*)$  for  $\omega^* \in \mathbb{Z}_{pq}^\times$ . In each game ‘‘Game  $i$ ’’, let  $S_i$  be the event that the attacker correctly guesses the hidden bit  $b$ . Without loss of generality, we assume that the adversary does not ask the same query more than once to the decapsulation oracle.

**Game 0:** This is the original attack game as defined in Definition 4.9. Thus, we have

$$\epsilon_{\mathcal{A}_T} = |\Pr[S_0] - 1/2|. \quad (4.4)$$

**Game 1:** This game is the same as above, with the exception that the random oracles are simulated as follows<sup>6</sup>:

**KDF:** An initially empty list KList is prepared. Given a query  $x$ , the following steps are performed: If  $(x, K)$  is in KList, then return  $K$ . Otherwise randomly generate  $K'$ , a bit-string of length  $KLen$ , add  $(x, K')$  to KList, and return  $K'$ .

**H:** An initially empty list HList is prepared. Given a query  $(x, \tau)$ , the following steps are performed: If  $((x, \tau), hash)$  is in HList, then return  $hash$ . Otherwise, randomly generate  $hash'$ , a bit-string of length  $hashLen$ , add  $((x, \tau), hash')$  to HList, and return  $hash'$ .

From the ideal assumptions to the random oracles in Game 0, we deduce that Game 0 and Game 1 are perfectly indistinguishable. Hence, we conclude

$$\Pr[S_0] = \Pr[S_1]. \quad (4.5)$$

**Game 2:** This game is the same as above, with the only difference that the challenge  $\Psi^* := (y^*, c_2^*)$  is fixed at the beginning of the game, where  $c_2^*$  is a randomly generated bit-string of length  $hashLen$ . It is easy to see that Game 1 and Game 2 are indistinguishable from the adversary’s point of view if none of the following events takes place:

- The adversary ever queries the oracle  $H$  on  $\omega^*$ . We call this event  $Ask_2$ .
- The adversary ever queries the KDF oracle on  $x$  with  $KDF(x) = dk_b$ , where  $dk_b$  is the challenge key given to the adversary in Step 4. The probability of this event is upperbounded by  $q_K/KLen$ .
- The decapsulation oracle is queried on  $(\Psi^*, \tau)$ , where  $\tau$  is the tag on which the adversary wishes to be challenged. From the restrictions on the adversary, this is only allowed before Step 6. In this case, however,  $\Psi^*$  is hidden to the adversary and thus the probability of this event is upperbounded by  $q_D/(n + hashLen)$ .
- The decapsulation oracle is queried on  $(\Psi^*, \tau')$ , where  $\tau \neq \tau'$  and  $H(\omega^*, \tau') = c_2$  holds (*i. e.*,  $(\Psi^*, \tau')$  is a valid encapsulation). The probability of this event and  $\neg Ask_2$  is upperbounded by  $q_D/hashLen$ .

<sup>6</sup>This affects the adversary’s oracle queries as well as the generation of the challenge encapsulation.

As the challenge in Game 2 is independent from everything the adversary knows, we conclude

$$\Pr[S_2] = \frac{1}{2}. \quad (4.6)$$

To compare Game 1 and Game 2, we use the following Lemma from [Sho04b]:

**LEMMA 4.3 (DIFFERENCE LEMMA)** *Let  $A, B, F$  be events defined in some probability distribution, and suppose that  $A \wedge \neg F \iff B \wedge \neg F$ . Then  $|\Pr[A] - \Pr[B]| \leq \Pr[F]$ .*

From this lemma, we conclude

$$|\Pr[S_1] - \Pr[S_2]| \leq \Pr[Ask_2] + q_K/KLen + q_D/(n + hashLen) + q_D/hashLen. \quad (4.7)$$

**Game 3:** This game is the same as above, except that instead of having access to the “real” decapsulation oracle, the adversary’s queries are responded using the following simulation: If a query  $(\Psi, \tau)$  is given, then the following steps are performed: Parse  $\Psi$  to  $(c_1, c_2)$ . Search HList, if there is an entry  $((x, \tau), hash)$  with  $|x|_2 \leq rLen$  and  $h_{pq}(x) = c_1$ . If yes and if  $hash = c_2$ , then return  $KDF(x)$  (simulated as above). Otherwise return  $\perp$ .

Let  $E_3$  be the event that the adversary invokes the decapsulation oracle on a valid query  $((c_1, c_2), \tau)$  without  $h_{pq}^{-1}(c_1)$  having been asked to  $H$ . Obviously, we have  $\Pr[E_3] \leq q_D/hashLen$ . If  $\neg E_3$  is true, then Game 2 and Game 3 are perfectly indistinguishable from the adversary’s point of view. Define  $Ask_3$  as the event that the adversary ever queries  $H$  or  $KDF$  on  $\omega^*$ . Again using the Difference Lemma, we have

$$|\Pr[Ask_2] - \Pr[Ask_3]| \leq q_D/hashLen. \quad (4.8)$$

It remains to bound  $\Pr[Ask_3]$ . For that reason, we describe an adversary  $\mathcal{A}_{Fact}$  against the  $p^2q$  factorization problem with advantage  $\epsilon' \geq \Pr[Ask_3]$ . Adversary  $\mathcal{A}_{Fact}$  proceeds as follows: On input  $n$ ,  $\mathcal{A}_{Fact}$  chooses  $r \in \mathbb{Z}_n$  at random and computes  $y^* = r^n \bmod n$ . Then  $\mathcal{A}_{Fact}$  lets the adversary  $\mathcal{A}_T$  run Game 3 on the public key  $n$ . In Step 4, a randomly generated bit-string of length  $KLen$  is passed to  $\mathcal{A}_T$ . When  $\mathcal{A}_T$  invokes the encapsulation oracle in Step 5, the challenge  $\Psi^* := (y^*, c_2)$  with a randomly generated bit-string  $c_2$  of length  $hashLen$  is responded to  $\mathcal{A}_T$ . The oracle queries are responded by  $\mathcal{A}_{Fact}$  exactly as in Game 3 except the following modification of the  $H$  oracle:

**H:** Given a query  $(x, \tau)$ , the following steps are performed: If  $((x, \tau), hash)$  is in HList, then return  $hash$ . Otherwise compute  $\gcd(r - x, n)$ . If  $\gcd(r - x, n)$  is a non-trivial factor of  $n$ , return this factor and halt. Randomly generate  $hash'$ , a bit-string of length  $hashLen$ , add  $((x, \tau), hash')$  to HList, and return  $hash'$ .

It is obvious that  $\mathcal{A}_{Fact}$  perfectly simulates the attack environment of  $\mathcal{A}_T$  in Game 3 and finds a non-trivial factor of  $n$  with probability  $\epsilon' \geq \Pr[Ask_3](1 - 1/p)$  (The attack may fail if  $r < pq$  holds, which occurs with probability  $1/p$ . Also note that the events  $r < pq$  and  $Ask_3$  are independent, because the distributions of  $\{x^n \bmod n \mid n \in \mathbb{Z}_n^\times\}$  and  $\{x^n \bmod n \mid n \in \mathbb{Z}_{pq}^\times\}$  are identical.). Thus, we conclude

$$\Pr[Ask_3] \leq \frac{\epsilon'}{1 - 1/p}. \quad (4.9)$$

Finally, from the equations (4.4), (4.5), (4.6), (4.7), (4.8), and (4.9), the assertion follows.  $\square$

REMARK 4.6 The gcd-trick exploited in the proof above is due to Fujisaki [Fuj01]. The benefit here is that  $h_{pq}$  evaluations are replaced by cheaper gcd-computations.

### 4.4.3 The Proposed Hybrid Encryption Scheme

In [AGK05], the generic Tag-KEM/DEM framework is formally defined. To avoid lengthy recurrences, we do not review this framework, but we only describe the concrete hybrid encryption scheme that is obtained when combining our proposed Tag-KEM with an appropriate DEM. The interested reader is referred to [AGK05] for the general treatment. Let  $(\mathcal{E}^{sym}, \mathcal{D}^{sym})$  be any one-time secure symmetric cryptosystem.

**Key Generation:** The key generation is the same as in TKEM.Gen(.).

**Encryption and decryption** is performed as follows:

$$\begin{array}{l|l}
 \mathcal{E}_{pk}(m) : & \mathcal{D}_{sk}(\Psi, \tau) : \\
 \omega \leftarrow \{0, 1\}^{rLen} & \text{parse } \Psi \text{ to } (c_1, c_2) \\
 dk := \text{KDF}(\omega) & r := c_1^d \bmod pq \\
 \tau \leftarrow \mathcal{E}_{dk}^{sym}(m) & \text{if } |r|_2 > rLen \text{ or } H(r, \tau) \neq c_2 \text{ return } \perp \text{ and halt} \\
 \Psi := (\omega^n \bmod n, H(\omega, \tau)) & m := \mathcal{D}_{\text{KDF}(r)}^{sym}(\tau) \\
 \text{Return } (\Psi, \tau) & \text{Return } m
 \end{array}$$

Note that the DEM ciphertext of the message  $m$  encrypted with the encapsulated one-time key serves as the tag. Thus, non-malleability of the DEM part is intuitively fulfilled because a CCA-secure Tag-KEM provides integrity of the tag. From the results of [AGK05], we derive

**THEOREM 4.8** *If the  $p^2q$  Factorization Assumption holds and if  $(\mathcal{E}^{sym}, \mathcal{D}^{sym})$  is one-time secure, then the proposed hybrid encryption scheme is CCA-secure in the random oracle model. More precisely, we have  $\epsilon_{hy} \leq 2\epsilon_{KEM} + \epsilon_{DEM}$ , where  $\epsilon_{hy}$ ,  $\epsilon_{KEM}$  and  $\epsilon_{DEM}$  denote the maximum advantage of an attack against the CCA security of proposed hybrid encryption scheme, against the CCA security of our new Tag-KEM, and against the one-time security of  $(\mathcal{E}^{sym}, \mathcal{D}^{sym})$ , respectively.*

In the above theorem, CCA-security of hybrid encryption is defined in the standard sense specified in Definition 2.4. Note that the reduction to factoring is tight.

REMARK 4.7 Interestingly, the proposed hybrid encryption scheme is very similar to the scheme obtained from applying the REACT-conversion to  $h_{pq}$  (see Example 4.2). The only difference in the REACT-case is that the inputs of the hash function  $H$  would be  $m, \omega$  and  $c_1$ . This is a small disadvantage because it decreases the efficiency and it is necessary to recover  $m$  before the second integrity check can be performed. Note that reject-timing attacks like [ST03, Den02] are possible if the timing difference between two different reject events is too large. In this attack, the adversary can find the secret key if she is able to distinguish the two different kinds of rejections of invalid ciphertexts. However, this threat can be easily dealt with on the implementation level (for instance by using a flag). As the trapdoor function  $h_{pq}$  is deterministic, the REACT version of our scheme can be tightly reduced to factoring, too. In particular, this remark shows that the advantages of our scheme compared to the members of the EPOC family result from the better suitability of our trapdoor permutation, not from the use of the Tag-KEM/DEM framework.

#### 4.4.4 Comparison

In this section, we give a brief comparison of hybrid encryption schemes using the Okamoto-Uchiyama technique Okamoto-Uchiyama encryption scheme and our proposed hybrid encryption scheme. We choose the Okamoto-Uchiyama trapdoor mechanism as a candidate, because like ours it is based on the  $p^2q$  Factorization Assumption. In Example 4.3, we have already introduced EPOC-3, which is the outcome of applying the REACT conversion to Okamoto-Uchiyama encryption.

A further well-known conversion technique has been proposed by Fujisaki and Okamoto in 1999 [FO99b]. It can be applied to any one-way function and one-time secure symmetric encryption scheme. Again, the resulting hybrid encryption scheme is IND-CCA in the ROM. The instantiation of this technique with Okamoto-Uchiyama encryption is denoted EPOC-2. In contrast to EPOC-3, EPOC-2 is CCA secure under the  $p^2q$ -factoring assumption in the ROM. Although in general the security reduction of the Fujisaki-Okamoto conversion technique is not very tight, Fujisaki observed a tight reduction proof tailored to the special application EPOC-2 [Fuj01]. A disadvantage of EPOC-2 is that in the decryption phase a re-encryption is necessary as an integrity check. For efficiency reasons, this re-encryption is only performed modulo  $q$  instead modulo  $n$  (accepting a small error probability). Nevertheless, the decryption is less efficient than in case of EPOC-3. There is also a second drawback due to the re-encryption: poor implementation makes EPOC-2 vulnerable against reject-timing attacks [ST03, Den02] as described in Remark 4.7. Here, the two kinds of rejection events are the enciphered text not meeting the length restrictions on the one hand, or failure of the re-encryption test on the other hand. As the latter involves the costly public key operations and hence takes a suitable amount of time, careless implementation makes it possible for an adversary to distinguish between the two cases by measuring the time of rejection.

Table 4.3 summarizes the most important parameters regarding security and performance. The efficiency of encryption and decryption is measured in modular multiplications, where  $MM(k)$  denotes a modular multiplication modulo a  $k$ -bit number. We do not distinguish between multiplications and squarings, and we assume that a modular exponentiation with a  $k$  bit exponent takes approximately  $3k/2$  modular multiplications, whilst a double exponentiation as necessary for performing the Okamoto-Uchiyama encryption Okamoto-Uchiyama encryption scheme takes approximately  $7k/4$  modular multiplications using standard techniques. We have not considered exponent recoding techniques, which are applicable in our scheme due to the fixed exponents. Chinese remaindering is taken into account if possible. Hashing, evaluations of the key derivation function and the symmetric key operations are not measured, because these magnitudes are comparable in all schemes. For evaluating the public key sizes, we compare  $n, g, h$  on the EPOC-2/3 side with  $n$  in our proposed scheme. The secret key sizes are the same ( $p, g_p$  for EPOC-2/3, resp.  $p, d$  for our proposed scheme). All quantities are measured in terms of the security parameter  $k$  (*i. e.*, the bit-length of the prime factors  $p, q$ ). In case of EPOC-2/3, we assume that  $rLen = k$  and  $hashLen \geq 2k$  hold (these are the values determining the exponent sizes).

As modular multiplication is quadratic in the length of the modulus, we conclude that our scheme is the most efficient one in decryption, whilst in encryption it is slightly less efficient than EPOC-2/3. Furthermore, the public key is 3 times shorter in our proposed scheme, and the underlying security assumption is optimal (as it is the case for EPOC-2).

Scheme	Assumption	encrypt	decrypt	pk	sk
EPOC-2	<b>FACT</b>	$\geq 7k/2 \text{ MM}(3k)$	$\approx 3k/2 \text{ MM}(2k) + 7k/4 \text{ MM}(k)$	$9k$	$3k$
EPOC-3	Gap-HR	$\geq 7k/2 \text{ MM}(3k)$	$\approx 3k/2 \text{ MM}(2k)$	$9k$	$3k$
Proposed	<b>FACT</b>	$\approx 9k/2 \text{ MM}(3k)$	$\approx 3k \text{ MM}(k)$	<b><math>3k</math></b>	$3k$

Table 4.3: Comparison of hybrid encryption schemes

Another advantage of our scheme is the following: If one-time pad is used for the symmetric part, then the message length in our scheme is  $2k$  compared to  $k$  in EPOC-2/3. This is because the bandwidth of  $h_{pq}$  is twice as large as the bandwidth of the Okamoto-Uchiyama encryption.

## 4.5 Application to Fail-stop Signature Schemes

Digital signatures were introduced to replace handwritten signatures in the electronic world, *i. e.*, the aim of a digital signature is to authenticate authorship of the signed message. The security of traditional signature schemes relies on a computational assumption. Provided that this assumption holds, no one but the owner of the secret signing key should be able to produce signatures that can be verified using the corresponding verification key. But an adversary who breaks this assumption is able to sign any message of her choice, such that the forged signatures pass the verification test, and the legal signer Alice has no chance to convince the recipient Bob (or a judge) that the signature in question has not been created by herself. To overcome this problem, *fail-stop signature (FSS) schemes* were invented. In a FSS scheme, the signer is protected against computationally unbounded adversaries in the following sense: If the signer is given a forged signature (*i. e.*, a signature passing the verification test but not created by herself), then with overwhelming probability the signer is able to prove that the underlying computational assumption has been broken and the protocol is stopped (hence the name *fail-stop* signature). Of course, a signer who breaks the underlying problem herself may exploit this mechanism to produce signatures which she later proves to be forgeries, *i. e.*, she can sign messages and disavow her signatures later. Therefore, the security for the recipients of fail-stop signatures against a cheating signer is still based on computational assumptions, whereas the signer is unconditionally secure<sup>7</sup>. As a consequence, FSS schemes are particularly suitable in asymmetric constellations, where the recipient (*e. g.*, a bank) is assumed to be much more powerful than the signer (*e. g.*, a private customer).

### Previous Work

Fail-stop signature schemes have been invented by Pfitzmann and Waidner in 1989 [PW90]. Whilst the earliest solutions were rather inefficient (bit-wise signing), the first practical FSS scheme has been observed soon [vHP93]. Luckily, van Heijst, Pfitzmann and Pedersen were able to generalize the discrete logarithm-based construction from [vHP93] to a generic construction based on a new primitive, the so-called *bundling homomorphisms* [vHPP92]. All subsequent provably secure FSS schemes known today follow that design.

<sup>7</sup>Note that this situation is dual to ordinary signature schemes, where the recipients are unconditionally secure and the signer's security relies on a computational assumption.

In this section, we focus on FSS schemes where the underlying problem is related to the integer factorization problem. Unfortunately, while there is an efficient FSS scheme based on the discrete logarithm problem [vHP93], the situation regarding factorization based FSS schemes is less satisfying. In 1991, the first factorization based FSS scheme has been published [vHPP92] (see [PP97] for a revised version). This scheme – called vHPP scheme in the following – is based on the intractability of factoring Blum integers  $n = pq$  with  $p \not\equiv q \pmod{8}$  (see Section 4.5.1 for a review of this FSS scheme). Until today, it is unknown if factoring integers of this special form is as hard as factoring arbitrary RSA-moduli. In addition, this scheme is quite complicated and the structure is not a “natural” one (the construction is defined in a way that the proofs work, but looks cumbersome at the first sight). Nevertheless, the vHPP scheme is the only previously known provably secure FSS scheme that is based on the factorization assumption only. All other factorization related FSS schemes are based on stronger assumptions or insecure. The first of these is [SSNP99], it is based on the factorization assumption but unfortunately turned out to be not provably secure (see [SSN03]).

The scheme of Susilo *et al.* [SSNGS00] is based on the so-called strong factorization assumption, which states that it is hard to factor  $n = pq$  even if an element  $g \in \mathbb{Z}_P^\times$  (for  $P \in \mathcal{PRIMES}$  and  $n \mid P - 1$ ) of multiplicative order  $p$  is known. Note that this assumption is in fact stronger than the factoring assumption, because there is no proof that knowledge of  $g$  does not weaken the hardness of factoring. Indeed, this assumption is considered as “quite unnatural” by Victor Shoup [Sho99]. Moreover, a successful attack against the signer’s security was recently published by the author of this thesis [SS04].

The most recent factorization-related FSS scheme is [SSN03], which is in fact an analogon of the discrete logarithm scheme [vHP93]. The scheme from [vHP93] utilizes two primes  $p, q$  with  $q \mid p - 1$ , and its security is based on the subgroup discrete logarithm problem in a  $q$ -element subgroup of the group  $\mathbb{Z}_p^\times$ . In [SSN03], the only difference is that the group  $\mathbb{Z}_p^\times$  is replaced by  $\mathbb{Z}_n^\times$ , where  $n$  is an RSA modulus. Consequently, this scheme is based on the subgroup discrete logarithm problem, too, and not on factoring as stated. In particular, there is no reduction that breaking this scheme enables to factor  $n$ . The only connection to factoring is that knowledge of the factors of  $n$  may weaken the schemes security, but this is of course not the meaning of “factorization-based”.

## Our contributions

Our major aim in this section is the development of a new factorization-based FSS scheme. The proposed scheme is the first one based on the intractability of factoring integers of  $p^2q$  type. As in terms of computational complexity both schemes are comparable, our new scheme provides a good alternative to the vHPP scheme.

Secondly, we point out that the generic construction using bundling homomorphisms is too restrictive. Namely, we show that a natural extension of our novel scheme formally does not meet the requirements demanded by van Heijst, Pedersen and Pfitzmann. These requirements are sufficient to prove security, but—as we will see—they are not necessary. Therefore, we propose an adjusted formulation of the general construction with relaxed requirements still sufficient for security. These considerations may be of independent interest, as, *e. g.*, they also lead to a fix of the recently broken scheme [SSNGS00]. We finally are able to show that the extension of our proposed basic scheme compares favorably with the vHPP scheme.

### 4.5.1 Basic Notions and Definitions for Fail-stop Signature Schemes

In this section, we briefly review the basic definitions related to FSS schemes. As ordinary signature schemes, FSS schemes consist of algorithms for key-generation, signing and signature testing. A signature passing the signature test is called *acceptable signature* in the following. Furthermore, to achieve the above mentioned extended security for the signer, there is an algorithm for proving that a forgery has occurred and an algorithm for verifying forgery proofs.

**DEFINITION 4.10 (FAIL-STOP SIGNATURE SCHEME)** *A fail-stop signature scheme is a quintuple  $(\text{KeyGen}, \mathcal{S}, \mathcal{V}, \text{Proof}, \text{Test})$  of polynomial time algorithms such that  $(\text{KeyGen}, \mathcal{S}, \mathcal{V})$  is a digital signature scheme as specified in Definition 2.5 and, in addition, the following holds:*

1. *Proof is a deterministic algorithm for proving forgeries. The input of Proof is a (presumably forged) acceptable signature/message pair and the signing key  $\text{sk}$ , the output is a bit-string proof.*
2. *Test is a deterministic algorithm which on input the public verification key and a bit-string proof outputs “valid” or “invalid”, indicating if proof is accepted as a proof of forgery or not.*

A secure FSS scheme has to fulfill two different security properties (for a more formal treatment see [PP97, PW90]):

- If an adversary knowing the public key and one correctly generated signature creates an acceptable signature, then with overwhelming probability the legal signer is able to present a valid proof of forgery. This property is referred to as *security for the signer* and it is not based on a computational assumption.
- A computationally bounded signer should not be able to create signatures that she later can prove to be forgeries. This property is referred to as *security for the recipients*, and it relies on the scheme’s underlying hard problem.

In this section, we call a forged signature that can be proved to be a forgery a *provable forgery*. The two security requirements have to be understood as independent and hence there are two different security parameters  $\gamma$  and  $k$ , related to the signer’s and the recipients’ security, respectively. The success probability of an unbounded adversary to create non-provable forgeries is upper-bounded by  $2^{-\gamma}$ , whereas the advantage of a cheating signer is a negligible function in  $k$ . Note that besides the possibility of proving forgeries, a signature scheme where forging signatures is easy does not make sense. Fortunately, it is proved in [PP97] that the above security requirements already imply that FSS schemes meet the strongest notion of security related to traditional signature schemes: existential unforgeability under adaptive chosen message attacks as specified in Definition 2.6. This proof can be sketched as follows: Assume that  $\mathcal{A}$  is a successful attacker against IND-CCA. Then Alice plays GAME.IND with  $\mathcal{A}$ , responds all signature queries using her secret key, and eventually,  $\mathcal{A}$  returns a new acceptable message/signature pair. The security for the signer implies that Alice can construct a valid proof of forgery for this pair with overwhelming probability. Hence, Alice is able to publish this signature and to

disavow it later, contradicting the security for the recipients. Note, however, that the basic FSS schemes we are analyzing in this section are only *one-time secure*, meaning that the number of the adversary's signature queries is restricted to one. In particular, this implies that the signer has to generate a fresh key pair for any new signature. It is possible to extend this variant to sign multiple messages [vHP93, BP97].

Another consequence of the signer's ability of disavowing forged signatures is that the key generation becomes slightly more complex than in ordinary signature schemes. In ordinary signature schemes, the key generation usually is a two-party protocol between the signer and a center<sup>8</sup>. In FSS schemes, a good key must guarantee both, the signer's as well as the recipients' security. Therefore, it is necessary that the recipient (or a third party trusted by the recipient) is involved in the key generating process. For simplicity, we only speak of a *center* in the following, capturing the cases that the center is a trusted third party, a recipient or a risk-bearer like an insurance that suffers damages if the recipient accepts invalid signatures. It is also possible to extend this model to multiple recipients (see [PP97]). Hence, in general the key generation is again a two party protocol between the signer and a center.

To simplify the situation with several signers, we only consider *FSS schemes with pre-key*. In this model, first the center generates a pre-key on his own and publishes it. Then each signer carries out a two-party protocol with the center to verify that the pre-key is "good" and finally, each signer creates her key-pair consisting of public/private key individually and publishes the public key. Note that there are general methods of verifying a pre-key that work independent from particular FSS schemes [PW90]. Therefore, a description of the pre-key verification protocol may be omitted when specifying a concrete FSS scheme.

### The General Construction Using Bundling Homomorphisms

In this section, we review a method of constructing FSS schemes with pre-key from any family of bundling homomorphisms proposed by van Heijst, Pfitzmann, and Pedersen [vHPP92].

Bundling homomorphisms can be understood as a special kind of collision-resistant hash functions.

**DEFINITION 4.11 (FAMILY OF BUNDLING HOMOMORPHISMS)** *Let  $I$  be a set of indices such that for each  $i \in I$   $(G_i, +, 0)$  and  $(H_i, *, 1)$  are finite Abelian groups. We define  $\mathcal{G} = (G_i, +, 0)_{i \in I}$  and  $\mathcal{H} = (H_i, *, 1)_{i \in I}$ . Let  $\mathcal{B} = \{h_i \mid h_i : G_i \rightarrow H_i\}_{i \in I}$  be a family of functions. Then  $\mathcal{B}$  is said to be a family of bundling homomorphisms if*

1. *There exists a probabilistic polynomial time key generator  $g$  which on input  $(\tau, k)$  outputs an index  $i \in I$  specifying  $h_i : G_i \rightarrow H_i$ .*
2. *For each  $i \in I$ , the function  $h_i : G_i \rightarrow H_i$  is a group homomorphism.*
3. *For each  $i \in I$ , each  $y \in \text{im}(h_i) \subseteq H_i$  has at least  $2^\tau$  pre-images in  $G_i$ .  $2^\tau$  is called the bundling degree.*

---

<sup>8</sup>The center has to convince the signer that the offered public key is indeed an outcome of KeyGen.

4. The members of  $\mathcal{B}$  are collision-resistant: For every probabilistic polynomial time algorithm  $\mathcal{A}$ , the following is negligible in  $k$ :

$$\Pr[i \leftarrow g(\tau, k); (x, \tilde{x}) \leftarrow \mathcal{A}(i) : \tilde{x} \neq x \text{ and } h_i(\tilde{x}) = h_i(x)].$$

5. The elements of  $\mathcal{B}$  are easy to evaluate: There exists a deterministic polynomial time evaluator  $\text{Eval}$  that on input  $i \in I, x \in G_i$  outputs  $h_i(x)$ .
6. For each  $i \in I$ , there must be polynomial time algorithms for
- computing the group operations in  $G_i$  and  $H_i$ ,
  - testing membership in  $G_i$  and  $H_i$ , and
  - selecting elements of  $G_i$  uniformly at random.

For  $i \leftarrow \text{KeyGen}(\tau, k)$ , we call  $h_i$  a  $(\tau, k)$ -bundling homomorphism.

For better readability, we omit the subscript  $i$  whenever it is clear from the context. Families of bundling homomorphisms can be used to construct provably secure FSS schemes with pre-key as follows:

Let  $\mathcal{B}$  be a family of bundling homomorphisms with key generating function  $g$  and let  $\gamma, k$  be two FSS security parameters. Then the components of a FSS scheme that is provably secure according to  $\gamma$  and  $k$  are given as:

**KeyGen:** Define  $\tau$  according to  $\gamma$  (details are given later). Then run  $g$  on  $(\tau, k)$  to obtain a  $(\tau, k)$ -bundling homomorphism  $h : G \rightarrow H$ . The groups  $G, H$  and  $h$  will serve as the pre-key.

For pre-key verification, the signer has to be convinced that  $h$  is a group homomorphism with bundling degree  $2^\tau$  (e. g., using a zero-knowledge proof<sup>9</sup>).

Finally, the signer chooses two elements  $\text{sk}_1, \text{sk}_2$  uniformly at random from  $G$  and computes  $\text{pk}_i = h(\text{sk}_i), i = 1, 2$ . This determines the secret key  $\text{sk} = (\text{sk}_1, \text{sk}_2)$  and the public key  $\text{pk} = (\text{pk}_1, \text{pk}_2)$ .

**S:** The message space  $\mathcal{M}$  is a suitable subset of  $\mathbb{Z}$ . To sign a message  $m \in \mathcal{M}$ , the signer computes

$$\mathcal{S}_{\text{sk}}(m) := \text{sk}_1 + m \text{sk}_2,$$

where  $m \text{sk}_2$  has to be understood as applying  $m$  times the group operation in  $G$  on  $\text{sk}_2$ .

**V:** An element  $\sigma \in G$  is an acceptable signature on  $m \in \mathcal{M}$  iff  $h(\sigma) = \text{pk}_1 * \text{pk}_2^m$  holds, where  $\text{pk}_2^m$  has to be understood as applying  $m$  times the group operation in  $H$  on  $\text{pk}_2$ .

*Proof:* Assume that  $\sigma^*$  is an acceptable signature on  $m$  which the signer wants to prove to be a forgery. To do so, the signer computes her own signature  $\sigma = \text{sk}_1 + m \text{sk}_2$  on  $m$ . If  $\sigma = \sigma^*$  holds, then the proof of forgery fails; otherwise, she presents proof =  $(\sigma, \sigma^*)$  as the proof of forgery.

---

<sup>9</sup>Note that there is no need to prove the collision-resistance of  $h$ , because the signer's security does not rely on this property.

*Test:* A pair  $(x, x') \in G \times G$  forms a valid proof of forgery iff  $x \neq x'$  and  $h(x) = h(x')$  hold.

Note that because of the homomorphic properties of  $h$ , each correctly generated signature passes the signature verification test. Following the above construction, the collision resistance of the bundling homomorphism can obviously be reduced to the security for the recipients. Indeed, if a signer Alice can construct any valid proof of forgery, then she can present this proof as a collision of the bundling homomorphism.

We now turn to the security for the signer. Figure 4.1 demonstrates the idea behind the generic construction.

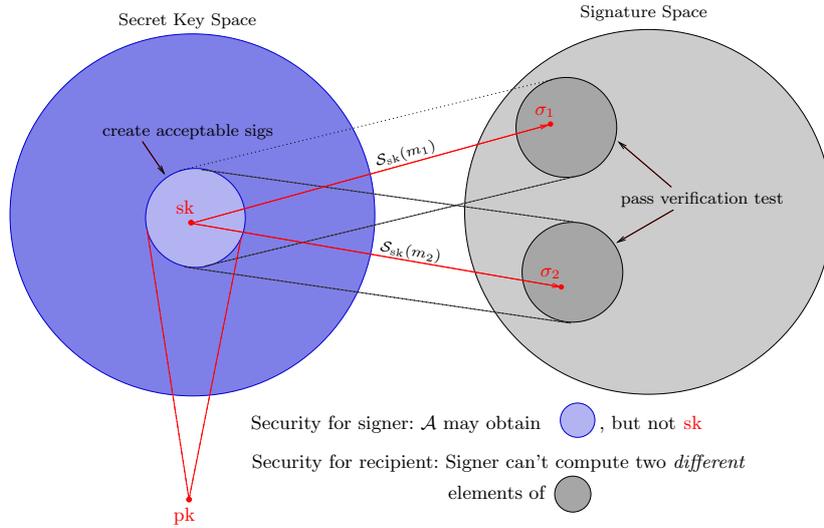


Figure 4.1: Scheme of Fail-stop Signatures

Assume that a signer Alice and a center follow the general construction. The crucial point is that because of property 3, Definition 4.11, there are at least  $2^{2\tau}$  possible secret keys  $sk' = (sk'_1, sk'_2)$  matching Alice's public key  $pk$  (in the sense that  $h(sk'_i) = pk_i, i = 1, 2$ ). Each of these keys produces acceptable signatures. Therefore, an adversary  $\mathcal{A}$  with unbounded computational power may be able to invert  $h$  and to find secret keys matching Alice's public key, but  $\mathcal{A}$  does not know which of the  $2^{2\tau}$  possibilities in fact equals Alice's secret key. However, the knowledge of a signature/message pair  $(\sigma, m)$  correctly generated by Alice provides  $\mathcal{A}$  with some extra information. In particular, as the equation  $sk_1 = \sigma - m sk_2$  must hold in  $G$ , the number of possible secret keys reduces to  $2^\tau$ . Alice is able to present a valid proof of forgery if the forged signature on a message  $m^*$  differs from her own signature on  $m^*$ , namely  $\sigma = sk_1 + m^* sk_2$ . Consequently, to measure  $\mathcal{A}$ 's probability of generating an unprovable forgery, we must find out how many of these possible keys produce the signature  $\sigma$  on  $m^*$ . As some easy implications show (see [PP97]), this number is upper-bounded by the magnitude of the set

$$T := \{d \in G \mid h(d) = 1 \wedge (m - m^*)d = 0\} \quad (4.10)$$

In order to upper-bound  $\mathcal{A}$ 's success probability, we have to consider the worst case, *i. e.*, we must find the maximum number taken over all possible messages  $m^* \neq m$ . Hence, we obtain the following bound:

$$T_{max} := \max_{0 \neq m' \in \mathcal{M}} \#\{d \in G \mid h(d) = 1 \wedge m'd = 0\}, \quad (4.11)$$

where  $m'$  is substituted for  $m - m^*$ . Concluding, we have shown that a computationally unbounded adversary  $\mathcal{A}$  can construct  $2^\tau$  possible secret keys of which at most  $T_{max}$  enable  $\mathcal{A}$  to produce *unprovable* forgeries.

These considerations lead to the following theorem (see [PP97] for a comprehensive proof in formal terms):

**THEOREM 4.9 (SECURITY OF THE GENERAL CONSTRUCTION)** *Let  $\gamma, k$  be security parameters and let  $\mathcal{B}$  be a family of bundling homomorphisms. Let  $\mathcal{F}$  be a FSS scheme following the general construction above. Then we have*

1.  $\mathcal{F}$  is  $k$ -secure for the recipients.
2. If the bundling degree  $2^\tau$  is chosen such that  $T_{max}/2^\tau \leq 2^{-\gamma}$ , then  $\mathcal{F}$  is  $\gamma$ -secure for the signer.

Consequently, the general construction offers a convenient tool for designing FSS schemes. Actually, to describe a FSS scheme based on the general construction, it is sufficient to specify a family of bundling homomorphisms and to determine the message space, the bundling degree and the number  $T_{max}/2^\tau$ . In particular, the scheme's underlying hard problem equals the problem of collision-finding in the family of bundling homomorphisms. The above construction is the basis of all previously known provably secure FSS schemes [PP97, SSNGS00, SSN03].

**EXAMPLE 4.4 (THE VHPP BUNDLING HOMOMORPHISM)** The following bundling homomorphism is the core of the previous factorization based fail-stop signature scheme due to van Heijst, Pedersen, and Pfitzmann [vHPP92]. The foundation of this approach is the concept of claw-free permutations, which was introduced by Goldwasser, Micali and Rivest in 1988 [GMR88].

Let  $\gamma, k$  be the security parameters related to the signer's and the recipients' security, respectively. Define the bundling degree  $\tau := \gamma + \rho$ , where  $\mathcal{M} := \{0, 1, \dots, 2^\rho - 1\}$  is the message space. On the input  $(\gamma, \tau)$ , the key generating function  $g$  chooses  $p, q \in \mathcal{PRIMES}(\lfloor k/2 \rfloor)$  with  $p = q = 3 \pmod{4}$  and  $p \not\equiv q \pmod{8}$ . Then the Abelian groups  $G$  and  $H$  are defined as follows:

$$G := (\mathbb{Z}_{2^\rho} \times (\pm \text{QR}(n))/\{1, -1\}, \circ, (0, 1)), \quad H := ((\pm \text{QR}(n))/\{1, -1\}, *, 1),$$

where the group operation  $\circ$  on  $G$  is given as

$$(a, x) \circ (b, y) := ((a + b \pmod{2^\tau}, xy4^{(a+b) \div 2^\tau}).$$

Each element of  $H$  is a coset  $\{x, -x\}$ , which is identified with its smaller member (*i. e.*, with  $x$ , if  $0 \leq x \leq (n-1)/2$ , and with  $-x$ , otherwise).

Finally, the bundling homomorphism  $h$  is defined by:

$$\begin{aligned} h : \mathbb{Z}_{2^\rho} \times (\pm \text{QR}(n))/\{1, -1\} &\longrightarrow (\pm \text{QR}(n))/\{1, -1\} \\ (a, x) &\mapsto \pm(4^a x^{2^\tau}) \pmod{n}, \end{aligned}$$

where again the notation  $\pm x$  in the image indicates that the coset  $\{x, -x\}$  is identified with its smaller member.

It can be shown that the above construction is a family of bundling homomorphisms under the assumption that factoring Blum integers  $n = pq$  with  $p \not\equiv q \pmod{8}$  is infeasible [PP97, vHPP92]. Note that this bundling homomorphism is quite artificial, namely the cumbersome group operation  $\circ$  in  $G$  is only chosen in order to provide  $h$  with homomorphic properties. Concerning the groups  $G$  and  $H$ , there are two reasons for considering the factor group modulo  $\{1, -1\}$  instead of  $\text{QR}(n)$ . On one hand, this choice anticipates the trivial collisions  $x^2 = (-x)^2 \pmod{n}$ , and on the other hand, it guarantees efficient membership testing in  $H$  (and hence in  $G$ )<sup>10</sup>.

### 4.5.2 The Proposed Fail-stop Signature Scheme

In this section, we introduce a new factorization-based FSS scheme and provide a complete security proof. We claim that the proposed construction is more simple and elemental than the artificial construction defining the vHPP scheme. The basic variant of our proposed scheme is an instance of the general construction described in Section 4.5.1. We will also introduce a natural extension of the basic version, which compares favorably with the vHPP scheme. Unfortunately, the generic security proof provided by the bundling homomorphism framework does not transfer completely to this extension. We will solve this problem by adjusting the generic construction in a suitable way. With this adjustment in mind, it is also possible to repair the recently broken FSS scheme from [SSNGS00].

#### The Basic Scheme

For the sake of completeness, we give the full description of our proposed scheme in the one recipient model. For the extensions to multiple recipients see [PP97].

**KeyGen** : On the input  $(\gamma, k)$ , the center defines  $\tau := \max(\gamma, \lfloor k/3 \rfloor)$  and chooses  $p, q \in \text{PRIMES}(\tau)$  with  $p \nmid q-1, q \nmid p-1, p \neq q$ . The Abelian groups according to  $n = p^2q$  are given as

$$G = H = (\mathbb{Z}_n^\times, *, 1).$$

The bundling homomorphism  $h$  is defined by

$$\begin{aligned} h : \mathbb{Z}_n^\times &\longrightarrow \mathbb{Z}_n^\times \\ x &\mapsto x^n \pmod{n} \end{aligned}$$

Let  $l$  be chosen such that  $2^l < p < 2^{l+1}$  holds. The groups  $G, H$ , the homomorphism  $h$ , and  $l$  will serve as the pre-key.

For pre-key verification, it is sufficient to assure the signer that  $n$  possesses a squared factor of approximate bit-length  $\gamma$  (e. g., using a zero-knowledge proof).

Finally, the signer selects two elements  $\text{sk}_1, \text{sk}_2 \in \mathbb{Z}_n^\times$  uniformly at random and computes  $\text{pk}_i = \text{sk}_i^n \pmod{n}, i = 1, 2$ . This determines the secret key  $\text{sk} = (\text{sk}_1, \text{sk}_2)$  and the public key  $\text{pk} = (\text{pk}_1, \text{pk}_2)$ .

<sup>10</sup>A number  $0 \leq x \leq (n-1)/2$  belongs to  $H$  iff the Jacobi symbol  $(\frac{x}{n})$  equals 1.

$\mathcal{S}$  : The message space is defined as  $\mathcal{M} = \{0, 1, \dots, 2^l\}$ . To sign a message  $m \in \mathcal{M}$ , the signer computes

$$\mathcal{S}_{\text{sk}}(m) := \text{sk}_1 * \text{sk}_2^m \bmod n.$$

$\mathcal{V}$  : An element  $\sigma \in \mathbb{Z}_n^\times$  is an acceptable signature on  $m \in \mathcal{M}$  iff  $\sigma^n = \text{pk}_1 * \text{pk}_2^m \bmod n$  holds.

*Proof* : Assume that  $\sigma^*$  is an acceptable signature on  $m^*$  which the signer wants to prove to be a forgery. To do so, the signer computes her own signature  $\sigma = \text{sk}_1 * \text{sk}_2^{m^*} \bmod n$  on  $m^*$ . If  $\sigma \neq \sigma^*$  holds, then the proof of forgery consists of  $(\sigma, \sigma^*)$ . The proof of forgery fails, otherwise.

*Test* : A pair  $(x, x') \in \mathbb{Z}_n^\times \times \mathbb{Z}_n^\times$  forms a valid proof of forgery iff  $x \neq x'$  and  $x^n = x'^n \bmod n$  hold.

This construction follows the general construction introduced in Section 4.5.1. Indeed, we can prove the following theorem:

**THEOREM 4.10** *Let  $I = \{n = p^2q \mid p, q \in \mathcal{PRIMES}, |p|_2 = |q|_2, p \nmid q - 1, q \nmid p - 1, p \neq q\}$  be a set of indices. Under the  $p^2q$  Factorization Assumption, the collection  $\mathcal{B} = \{h^{(n)}\}_{n \in I}$  is a family of bundling homomorphisms, where  $h^{(n)}$  is defined as*

$$\begin{aligned} h^{(n)} : \mathbb{Z}_n^\times &\longrightarrow \text{N-R}(n) \\ x &\mapsto x^n \bmod n. \end{aligned}$$

**PROOF** We have already proved in Corollary 4.1 that  $h$  is *p-to-one* and in Corollary 4.2 that  $h$  is collision-resistant under the  $p^2q$  Factorization Assumption. Thus the assertion follows, as the remaining requirements are obviously fulfilled.  $\square$

Theorem 4.10 implies the first part of the security proof for the new scheme:

**COROLLARY 4.3** *Under the  $p^2q$  Factorization Assumption, the proposed scheme as defined above is secure for the recipients.*

To complete the security proof, we show the following theorem:

**THEOREM 4.11** *The proposed scheme as defined above is secure for the signer.*

**PROOF** According to Theorem 4.9, we have to determine the number

$$T_{\max} := \max_{0 \neq m' \in \mathcal{M}} \#\{d \in G \mid h(d) = 1 \wedge m'd = 0\}$$

and show that  $T_{\max}/2^\tau \leq 2^{-\gamma}$  is fulfilled. Putting in the parameters of the proposed scheme, we obtain

$$T_{\max} = \max_{0 < m' < p} \#\{d \in \mathbb{Z}_n^\times \mid d^n = 1 \bmod n \wedge d^{m'} = 1 \bmod n\} = 1,$$

because  $d^n = 1 \bmod n$  implies  $d = 1 \vee \text{ord}_n(d) = p$  (see Lemma 4.1 and Lemma 4.2). Thus,  $d^{m'} = 1 \bmod n$  is fulfilled only for  $d = 1$ . Hence, we conclude  $T_{\max}/2^\tau \leq 2^{-\gamma}$ , because  $\tau$  was chosen as the maximum of  $\gamma$  and  $\lfloor k/3 \rfloor$ .  $\square$

### The Extended Scheme

The question arises why we don't use the natural message space  $\mathbb{Z}_n$ . Examining the preceding proof discloses that for  $\mathcal{M} = \mathbb{Z}_n$  the cardinality of  $T_{max}$  would be  $p$  instead of 1 : with the same reasoning as above, we have for  $m^* \in \mathbb{Z}_n$

$$\begin{aligned} p \mid m - m^* &\Rightarrow \#T = \#\{d \in G \mid d^n = 1 \pmod n \wedge d^{m-m^*} = 1\} & (4.12) \\ &= \#\{d \in G \mid \text{ord}_n(d) = p\} \\ &= p \end{aligned}$$

$$p \nmid m - m^* \Rightarrow \#T = \#\{d \in G \mid d^n = 1 \pmod n \wedge d^{m-m^*} = 1\} = 1, \quad (4.13)$$

where  $T$  is defined as in eq. 4.10. Recall that  $T$  upperbounds the number of possible secret keys (out of  $2^\tau$ ) that enable the adversary to produce unprovable signature forgeries on the message  $m^*$ . Summing up, if the message  $m^*$  that the adversary chooses to forge and the given message  $m$  are congruent modulo  $p$ , then the forgery proof constructed by the signer following the protocol described in Section 4.5.1 may fail with probability exceeding  $2^{-\gamma}$  (to be more precise, we can upperbound the adversary's advantage by  $p2^{-\gamma}$ ). On the other hand, if  $m^* = m \pmod p$  holds, then the signer is able to factor the modulus by computing  $p = \text{gcd}(n, m - m^*)$ . Thus, if we allow an alternative forgery proving mechanism—namely the presentation of two messages leading to a solution of the underlying hard problem—then we can extend the message space to  $\mathbb{Z}_n$ .

In the following, we explain how the general framework has to be adjusted to take the above considerations into account. Recall that FSS schemes are one-time signature schemes, *i. e.* the adversary has notice of one valid signature/message pair  $(m, \sigma)$  created by the legitimate signer. The crucial observation is that certain messages which the adversary may select for signature forging enable the signer to solve the underlying problem. That is, on these messages unprovable forgeries are a priori impossible. We call these messages *disclosing* according to  $m$ , and we denote its union with  $\mathcal{DC}(m)$ . Indeed, when determining the maximum amount of the set  $T$  from eq. (4.10), we only have to consider messages  $m^* \notin \mathcal{DC}(m)$ . Consequently, we redefine the set  $T_{max}$  as follows:

$$T_{max} := \max_{m \neq m^* \in \mathcal{M}; m^* \notin \mathcal{DC}(m)} \#\{d \in G \mid h(d) = 1 \wedge (m - m^*)d = 0\} \quad (4.14)$$

In case of our proposed scheme, eq. 4.13 shows that the definition  $\mathcal{DC}(m) = \{\tilde{m} \in \mathbb{Z}_n \mid m = \tilde{m} \pmod p\}$  suffices to provide security for the extended message space  $\mathbb{Z}_n$ . Indeed, we have  $T_{max} = 1$ , and consequently  $T_{max}/2^\tau \leq 2^{-\gamma}$  as before.

In addition, an alternative forgery proving mechanism for the general construction has to be specified as explained above. When presenting a proof of forgery, Alice indicates which of the two mechanisms she has used. In case of our proposed scheme, given a forgery  $(\sigma^*, m^*)$ , Alice proceeds as follows: first, she constructs her own signature  $\sigma$  on the message  $m^*$ . If  $\sigma \neq \sigma^*$  holds, as usually she presents the pair  $(\sigma, \sigma^*)$  as a proof of forgery. Otherwise, Alice indicates that she uses the alternative procedure and her proof of forgery consists of  $(m', m^*)$ , where  $m'$  is the message Alice has signed previously with her signing key  $sk$ . The latter proof is checked by verifying that  $m^* \in \mathcal{DC}(m')$  is fulfilled, or—in our concrete instantiation—that  $\text{gcd}(m' - m^*, n)$  is a non-trivial factor of  $n$ , respectively.

Note that this adjustment additionally requires that Alice keeps the signed message in memory. As the analyzed FSS schemes are one-time schemes, however, this is a quite

small additional overhead (Alice has to store the secret key for constructing original forgery proofs anyway).

REMARK 4.8 In [SSNGS00], the authors confused the sets  $T$  and  $T_{max}$  and incorrectly tried to prove the security of their scheme by showing that the cardinality of  $T$  is sufficiently small. As shown in [SS04], this flawed application of the generic construction leads to a successful attack against the signer’s security. With the same ideas as described above, it is possible to “re-prove” the security of the original scheme with an alternative forgery proof mechanism.

### 4.5.3 Comparison

Table 4.4 provides a detailed comparison of the vHPP scheme and the proposed ones. As usual,  $\gamma$  and  $k$  are the security parameters related to the signer’s and recipients’ security, respectively. As it is common in the literature about FSS schemes, we relate the parameters to the message length.

	vHPP Scheme	Proposed (basic var.)	Proposed (ext. var.)
Message length	$\rho$	$\rho = \max(\gamma, k/3)$	$\rho = \max(3\gamma, k)$
Sig. length	$\gamma + \rho + k$	$3\rho$	$\rho$
Length of pk	$2k$	$6\rho = \max(6\gamma, 2k)$	$2\rho = \max(6\gamma, 2k)$
Length of sk	$2(\gamma + \rho + k)$	$6\rho$	$2\rho$
Sign (#MM)	$\rho$	$\rho$	$\rho$
Verify (#MM)	$< 2\rho + \gamma$	$< 4\rho$	$< 4/3\rho$
Assumption	Fact. of $n = pq$ $p = q = 3 \bmod 4, p \neq q \bmod 8$	Fact. of $n = p^2q$	Fact. of $n = p^2q$

Table 4.4: Comparison of factorization-based fail-stop signature schemes

Due to the interaction of the different parameters, a general evaluation is difficult. As a rough guideline, in case of  $k > \gamma$ , the basic variant of the proposed scheme outperforms the vHPP scheme in most points, whereas in case of  $k < \gamma$  the vHPP scheme is more advantageous. But in both cases, the differences are not large. The extended variant of the proposed scheme, however, can be easily verified to be the best among the three.

In the discussion above, we compared the proposed schemes with the vHPP scheme, because these schemes are based on the factoring assumption. But it has to be mentioned that signature generation is even more efficient in van Heijst and Pedersen’s discrete logarithm scheme [vHP93]. This is due to the fact that the group  $G$  in this scheme is *additive*, and therefore modular exponentiation is replaced by modular multiplication. Therefore, important further work in this field is the development of a FSS scheme which is either: based on a fairly weak factorization assumption and as efficient as the discrete logarithm scheme.

## 4.6 Application to Trapdoor Commitment Schemes (Trapdoor Hash Families)

Commitment schemes are an important primitive in public key cryptography. Among these, *trapdoor* commitment schemes play a leading role, especially in the context of zero-

knowledge protocols.

Recently, trapdoor commitment schemes found new applications in a different setting, namely in designing digital signature schemes with additional properties, as there are on-line/off-line signatures, chameleon signatures and signcryption [ST01, KR00, ADR02]. We will sketch these applications in Section 4.6.2.

In this new context, trapdoor commitment schemes are not longer described as a protocol between the commiter and the receiver, but a functional oriented definition is more convenient, and the term trapdoor commitment scheme is replaced by trapdoor hash family, *a. k. a.* chameleon hash family. As pointed out by Krawczyk and Rabin [KR00], both definitions are equivalent (for *non-interactive* trapdoor commitment schemes).

#### 4.6.1 Basic Notions and Definitions for Trapdoor Hash Families

A trapdoor hash function is a special kind of hash function where no one else as the holder of a trapdoor is able to construct collisions.

**DEFINITION 4.12 (TRAPDOOR HASH FAMILY)** *A trapdoor hash family is a pair  $(\text{KeyGen}, \mathcal{H})$  of probabilistic polynomial time key generation algorithm and a family of polynomial time hash functions such that the following holds:*

**Key Generation:** *On input a security parameter  $1^k$ , the randomized algorithm  $\text{KeyGen}$  outputs a pair  $(\text{hk}, \text{tk})$  of hash and trapdoor key. The hash key  $\text{hk}$  uniquely specifies an element  $\text{hash}_{\text{hk}}$  of the family  $\mathcal{H}$ .*

**Hash:** *The algorithm  $\text{hash}_{\text{hk}} : \mathcal{M} \times \mathcal{R} \rightarrow \mathcal{H}$  computes the hash value, where  $\mathcal{M}$  and  $\mathcal{R}$  are the message space and the space of randomness, respectively.*

**Weak Altering:** *For each trapdoor key  $\text{tk}$  there is a polynomial time algorithm  $w\text{Alt}_{\text{tk}} : \mathcal{M} \times \mathcal{R} \times \mathcal{M} \rightarrow \mathcal{R}$  which given a message  $m$ , randomness  $r$ , and a target message  $m'$  computes randomness  $r'$  with  $\text{hash}_{\text{hk}}(m; r) = \text{hash}_{\text{hk}}(m'; r')$ . The pair  $((m, r), (m', r'))$  is called a trapdoor collision.*

The algorithm  $w\text{Alt}_{\text{tk}}$  enables the owner of the trapdoor key to compute a hash value  $\text{hash}_{\text{hk}}(m; r) = h$  and “alter” the meaning of  $h$  in any desired way (*i. e.*, he can claim that  $h$  is the hash value of an arbitrary message  $m' \in \mathcal{M}$  by presenting the randomness  $r'$  with  $\text{hash}_{\text{hk}}(m'; r') = h$  as a proof). For some applications, however, a strictly stronger property turns out to be useful, namely the owner of the trapdoor key should be able to alter a hash value arbitrarily even without knowledge of the pre-image values  $r, m$ . We call this property *strong altering*<sup>11</sup>:

**DEFINITION 4.13 (STRONG ALTERING)** *A trapdoor hash family  $(\text{KeyGen}, \mathcal{H})$  provides strong altering if for each key  $(\text{hk}, \text{tk}) \leftarrow \text{KeyGen}(1^k)$  there is a polynomial time algorithm  $s\text{Alt}_{\text{tk}} : \mathcal{H} \times \mathcal{M} \rightarrow \mathcal{R}$  with the following property: Given a hash value  $h \in \mathcal{H}$  and a target message  $m \in \mathcal{M}$ , the algorithm  $s\text{Alt}_{\text{tk}}$  computes randomness  $r \in \mathcal{R}$  such that  $h = \text{hash}_{\text{hk}}(m; r)$ .*

<sup>11</sup>In [ST01], this property is referred to as *inversion property*.

The security of a trapdoor hash family requires that without knowledge of the trapdoor key it should be hard to find collisions. Moreover, the hash pairs obtained by invoking the altering algorithm should be indistinguishable from “real” hash pairs.

**DEFINITION 4.14 (SECURITY OF TRAPDOOR HASH FAMILIES)** *The trapdoor hash family  $(\text{KeyGen}, H)$  is secure if the following properties hold:*

**Collision resistance:** *For any PPA  $\mathcal{A}$  the following probability is negligible in  $k$ :*

$$\Pr[(\text{hk}, \text{tk}) \leftarrow \text{KeyGen}(k), (r, m, r', m') \leftarrow \mathcal{A}(\text{hk}) : \\ (r, m) \neq (r', m') \wedge \text{hash}_{\text{hk}}(m; r) = \text{hash}_{\text{hk}}(m'; r')]$$

**Uniformity:** *The outcome of  $s/w \text{Alt}_{\text{tk}}$  is uniformly distributed in  $\mathcal{R}$  provided that the randomness input is also uniformly distributed in  $\mathcal{R}$ .*

## 4.6.2 Recent Applications for Trapdoor Hash Families

### On-line/Off-line Signatures

A digital signature scheme possesses the *on-line/off-line property* if the signer is able to perform the bulk of the computation off-line, *i. e.*, before receiving the message that has to be signed. In [ST01], Shamir and Tauman proposed a generic construction for transforming a weakly secure signature scheme into a strong on-line/off-line signature scheme improving an early proposal of Even, Goldreich and Micali [EGM96]. Informally, their idea is the following: The signer runs two key generation algorithms: for an ordinary signature scheme as well as for a trapdoor hash family (in particular, the signer holds the hash trapdoor). In the off-line phase, the signer randomly selects  $r'$  and  $m'$  and constructs the dummy hash value  $h = \text{hash}_{\text{hk}}(m'; r')$ . She signs  $h$  and keeps the obtained signature  $\sigma$  and the values  $r', m', h$  in memory. In the on-line phase, when the message  $m$  that has to be signed is known, the signer invokes the altering algorithm to obtain randomness  $r$  with  $\text{hash}_{\text{hk}}(m; r) = h$ . The tuple  $(r, \sigma)$  then defines the signature of  $m$ . Signature verification is straight-forward, as by construction  $\sigma$  is a valid hash-then-sign signature of  $m$ . Note that only for executing the protocol weak altering is sufficient. The strong altering property, however, leads to more powerful constructions (see Section 2.3 for security notions related to digital signature schemes): instantiated with a signature scheme that is existentially unforgeable under generic chosen message attacks, the above construction (with a weak hash family) is secure against adaptive chosen message attacks (EF-CMA). If strong altering is possible, the original signature scheme only has to be unforgeable against known-message attacks to ensure EF-CMA security of the converted online-offline scheme. To sum up, we are looking for a trapdoor hash family where strong altering is possible and weak altering is fast. Some previous constructions of trapdoor hash families provide strong altering algorithms, but the weak altering is not particularly efficient [Gen04, FF02, KR00]. In contrast, Shamir and Tauman constructed a new trapdoor hash family with a very fast weak altering mechanism, but lacking the strong altering property. In the following, we will give the first trapdoor hash family that combines both properties: extremely fast weak altering and the possibility of strong altering.

## Signcryption

In [ADR02] An *et al.* give a formal treatment of securely carrying out joint encryption and signature in the public-key setting. Roughly speaking, the security goals here are to protect the sender’s authenticity and the receiver’s privacy. In addition to examining the classical “encrypt-then-sign” and “sign-then-encrypt” paradigms, An *et al.* propose a new variant which they call “commit-then-encrypt-and-sign” (*Ct&S*). In this setting, it is possible to perform the costly public-key operations (encrypt and sign) in parallel, therefore saving computation time<sup>12</sup>. For the sake of uniformity, we describe the basic procedure in hash-terminology: The sender hashes the message, then in parallel he signs the hash and encrypts the message. Finally he transmits the hash data, the signature and the ciphertext to the receiver. Decryption and verification is straight-forward. Until now, the trapdoor has not been exploited. The benefit of a trapdoor hash family arises if for signing Shamir-Tauman on-line/off-line signatures are used. Indeed, combined with an on-line/off-line encryption scheme (*e. g.*, hybrid encryption), a complete on-line/off-line signcryption scheme can be obtained as follows: In the off-line phase, the sender creates a fake hash  $\text{hash}_{\text{hk}}(m'; r')$ , builds the signature  $\sigma$  of  $\text{hash}_{\text{hk}}(m'; r')$  and performs the off-line encryption operations (*e. g.*, constructing and encapsulating a session key). Once the message  $m$  is known, the sender invokes the altering algorithm to obtain randomness  $r'$  with  $\text{hash}_{\text{hk}}(m'; r') = \text{hash}_{\text{hk}}(m; r)$ , completes the encryption (*e. g.*, encrypting  $m$  symmetrically with the session key) and finally transmits  $r$ ,  $\sigma$  and the obtained ciphertext  $c$ . The receiver encrypts  $c$ , gets  $m$  and verifies if  $\sigma$  is a valid signature of  $\text{hash}_{\text{hk}}(m; r)$ . It has not been pointed out by An *et al.* that this procedure even enhances the security of the original signature scheme. But along the lines of Shamir-Tauman’s security proof it can be easily shown that a signature scheme secure against generic message attacks is sufficient to receive an EF-CMA secure conversion. If the trapdoor hash family provides strong altering, even security against random message attacks suffices.

## Chameleon Signatures

The concept of chameleon signatures was introduced by Krawczyk and Rabin to simplify undeniable signatures [KR00]. The aim of chameleon signatures is to distract the receiver of a signature from revealing the signed message to any third party. To realize this, the well-known hash-then-sign paradigm reviewed in Section 2.3.3 is used, but the conventional hash function is replaced by a trapdoor hash. Here, only the receiver holds the secret trapdoor and is therefore able to forge valid signatures for messages of his choice by invoking the altering algorithm. Consequently, no third party will be convinced of the validity of any signature, because the receiver could have created it himself. On the other hand, the signer is protected against an dishonest receiver who creates a fake valid signature, because in this case the legitimate signer is able to present a collision of the trapdoor hash function with overwhelming probability. This enables the signer to deny the forged signature. In [CGHGN01], Catalano *et al.* pointed out that on-line/off-line chameleon signatures can be constructed when using an on-line/off-line trapdoor hash family.

<sup>12</sup>Note that the naive approach of just signing and encrypting a message in parallel and sending the signature along with the ciphertext may violate the receiver’s privacy, because the signature may reveal information about the message.

### 4.6.3 The Proposed Trapdoor Hash Families

#### The Proposed Strong Trapdoor Hash Family with Fast Altering.

The following trapdoor hash family with fast altering can be compared to Shamir-Tauman's fast altering trapdoor hash family from [ST01]. Shamir and Tauman utilize the homomorphic one-way function  $x \mapsto g^x \bmod n$ , where  $n$  is a safe RSA modulus and  $g$  is an element of maximal order  $\lambda(n)$  in  $\mathbb{Z}_n^\times$ . We will use the one-way homomorphism  $x \mapsto x^n \bmod n$  for  $n = p^2q$ . In both cases, the idea is that a non-trivial element of the function's kernel reveals the factorization of  $n$  (by providing a multiple of  $\lambda(n)$ , resp.  $pq$ ), which implies collision resistance. The main difference is that only in our case the function is trapdoor, which additionally provides the strong altering property.

**Key Generation:** Let  $k$  be a security parameter. On input  $1^k$ , choose  $p, q \in \text{PRIMES}(k)$  with  $p \nmid q - 1, q \nmid p - 1, p \neq q$ , and set  $n = p^2q$ . Define  $l$  such that  $2^l < pq < 2^{l+1}$  holds.

The hash key is  $\text{hk} = (n, l)$  and the trapdoor key is  $\text{tk} = (p, q)$ . The message space and the space of randomness are defined as  $\mathcal{M} = \{0, \dots, [n]^{k-l-1} - 1\}$  and  $\mathcal{R} = \mathbb{Z}_{pq}$ , respectively.

**Hash:** To hash a message  $m \in \mathcal{M}$ , a value  $r \in \mathbb{Z}_{pq}$  is chosen uniformly at random and  $\text{hash}_{\text{hk}}(m; r) = (m||r)^n \bmod n$  is computed, where  $m||r$  denotes the concatenation of  $m$  and  $r$ .

We can prove the following theorem:

**THEOREM 4.12** *Under the  $p^2q$  Factorization Assumption, the above construction is a secure trapdoor hash family with extremely fast weak altering and the strong altering property.*

- PROOF**
1. Extremely fast weak altering: From Theorem 4.1, we conclude that  $m' || r' = m || r \bmod pq$  is equivalent to  $\text{hash}_{\text{hk}}(m'; r') = \text{hash}_{\text{hk}}(m; r)$ , thus  $r = 2^{l+1}(m' - m) + r' \bmod pq$  yields the desired result.  $r$  can be computed extremely fast as multiplication with  $2^{l+1}$  is just a bit shift operation.
  2. Strong altering: Let  $h$  be a possible hash value and let  $m$  be the target message. From Theorem 4.1, we conclude that  $r = h^{1/n} - 2^{l+1}m \bmod pq$  leads to  $h = \text{hash}_{\text{hk}}(m; r) = (m || r)^n = (2^{l+1}m + r)^n \bmod n$ .
  3. Uniformity of altering: The considerations above show that for any hash value  $h$  and any message  $m$  there is a unique  $r \in \mathbb{Z}_{pq}$  with  $h = \text{hash}_{\text{hk}}(m; r)$ . Consequently, uniformity holds for both altering algorithms.
  4. Collision resistance under the  $p^2q$  Factorization Assumption: As  $\text{hash}_{\text{hk}}(m : r) = \text{hash}_{\text{hk}}(m'; r')$  is equivalent to  $(2^{l+1}m + r)^n = (2^{l+1}m' + r')^n \bmod n$ , we have  $2^{l+1}m + r = 2^{l+1}m' + r' \bmod pq$  using Theorem 4.1. Due to the length-restrictions of  $m$  and  $r$ , it is impossible that this equality holds modulo  $n$ , thus we must have  $\text{gcd}(m || r - m' || r', n) = pq$ .

□

REMARK 4.9 In terminology of commitment schemes, the above construction is a perfectly hiding and computationally binding trapdoor commitment scheme. We further want to point out that for the applications in on-line/off-line signature scheme, resp. signcryption, the sender is also the trapdoor holder. Therefore, it is not a problem for him to choose elements of  $\mathbb{Z}_{pq}$  uniformly at random. For more general applications, the randomness has to be taken from the set  $\{0, 1\}^l$  instead. In this case, however, the proposed scheme is not longer perfectly binding. We assume that binding holds at least computationally.

### The Proposed Strong Trapdoor Hash Family with On-Line/Off-Line Property.

**Key Generation:** Let  $k$  be a security parameter. On input  $1^k$ , choose  $p, q \in \mathcal{PRIMES}(k)$  with  $p \nmid q - 1, q \nmid p - 1, p \neq q$ , and set  $n = p^2q$ . Define  $l$  such that  $2^l < pq < 2^{l+1}$  holds.

The hash key is  $\text{hk} = (n, l)$ , and the trapdoor is the factorization of  $n$ . Set  $\mathcal{M} = \{0, \dots, [n]^{k-l-1} - 1\}$  and  $\mathcal{R} = \mathbb{Z}_{pq} \times \mathbb{Z}_n^\times$ .

**Hash:** To hash a message  $m \in \mathcal{M}$ , one chooses two random values  $s \in \mathbb{Z}_{pq}, r \in \mathbb{Z}_n^\times$  and computes  $\text{hash}_{\text{hk}}(m; r, s) = (1 + (m||s)n)r^n \bmod n^2$ , where  $m||s$  denotes the concatenation of  $m$  and  $s$ .

We have the following theorem:

THEOREM 4.13 *Under the  $p^2q$  Factorization Assumption the above construction is a secure trapdoor hash family with strong altering and on-line/off-line property.*

PROOF 1. Strong altering: Let  $h$  be a possible hash value and let  $m$  be the target message. We show how to construct randomness  $s \in \mathbb{Z}_{pq}, r \in \mathbb{Z}_n^\times$  with  $h = \text{hash}_{\text{hk}}(m; r, s) = (1 + (m||s)n)r^n = (1 + (2^{l+1}m + s)n)r^n \bmod n$ . From Theorem 4.4(3), we know that there are  $r_{pq} \in \mathbb{Z}_{pq}^\times, m_{pq} \in \mathbb{Z}_{pq}, 0 \leq i < p$  with  $h = (1 + (m_{pq} - ir_{pq}^{-1}pq)n)(r_{pq} + ipq)^n \bmod n^2$ . The values  $r_{pq}$  and  $m_{pq}$  can be computed from  $h$ :  $r_{pq} = h^{1/n} \bmod pq, m_{pq} = L_n(r_{pq}^{-n}h \bmod n^2) \bmod pq$ . Thus, to achieve  $h = \text{hash}_{\text{hk}}(m; r, s)$ , we have to find  $r \in \mathbb{Z}_n^\times, s \in \mathbb{Z}_{pq}, 0 \leq i < p$  with

$$2^{l+1}m + s = m_{pq} - ir_{pq}^{-1}pq \bmod n \quad (4.15)$$

$$r = r_{pq} + ipq \bmod n \quad (4.16)$$

The first equation (4.15) uniquely determines  $s \bmod pq$  and  $i$ . From eq. (4.16), one immediately computes  $r \bmod n$ .

2. Weak altering: From the considerations above we easily deduce the following procedure: Given  $m, m' \in \{0, \dots, [n]^{k-l-1} - 1\}, r \in \mathbb{Z}_n^\times, s \in \mathbb{Z}_{pq}$ , the weak altering algorithm computes  $r' \in \mathbb{Z}_n^\times, s' \in \mathbb{Z}_{pq}$  by solving the following system of modular equations:

$$\begin{aligned} 2^{l+1}m + s &= 2^{l+1}m' + s' - ir^{-1}pq \bmod n \\ r &= r' + ipq \bmod n \end{aligned}$$

3. Collision resistance under the  $p^2q$  Factorization Assumption: First, we observe that  $\text{hash}_{\text{hk}}(m; r, s) = \text{hash}_{\text{hk}}(m'; r', s')$  is equivalent to  $(1 + n(2^{l+1}m + s))r^n = (1 + n(2^{l+1}m' + s'))r'^n \pmod{n^2}$ . From Theorem 4.4(3), we therefore conclude

$$\begin{aligned} 2^{l+1}m + s &= 2^{l+1}m' + s' - ir^{-1}pq \pmod{n} \text{ for some } 0 \leq i < p \text{ and} \\ r &= r' + ipq \pmod{n} \end{aligned}$$

From the length restrictions of  $m, m', s$  and  $s'$  we deduce that  $i = 0$  is impossible. Thus, we can factor  $n$  by computing  $\text{gcd}(r - r', n) = pq$ .

4. Uniformity of altering: From the consideration about strong altering we conclude that for each message  $m \in \mathcal{M}$  and each hash value  $h$  there is exactly one  $s \in \mathbb{Z}_{pq}$  and one  $r \in \mathbb{Z}_n^\times$  satisfying  $\text{hash}_{\text{hk}}(m; r, s) = h$ . Thus, the uniformity property trivially holds for both altering algorithms.
5. On-line/off-line property: After the message  $m$  is retrieved, only one modular multiplication and one modular addition have to be performed, as the computation of  $r^n n \pmod{n^2}$  can be done in advance.

□

However, we still have to resolve a last problem: For sound execution the secret trapdoor has to be known to the user of the hash function, because he must choose  $s$  uniformly at random from  $\mathbb{Z}_{pq}$ . The same problem occurs in the factorization-based scheme from [BCP03], where the randomness has to be sampled from  $\mathbb{Z}_{n\lambda(n)/2}^\times$ . But unfortunately, in a chameleon hash signature scheme the trapdoor must be *unknown* to the signer/haser, as otherwise the signer is able obtain hash collisions and thus to deny her signatures. This difficulty could be overcome by providing the user an upper bound of the secret numbers  $pq$  resp.  $n\lambda(n)/2$ , but it has to be pointed out that proceeding in that way usually leads to a loss of uniformity. But luckily, it turns out that in our case the problem can be solved nevertheless: If the randomness  $s$  is sampled from the publicly known set  $\{0, 1\}^{l+1}$  instead of  $\mathbb{Z}_{pq}$ , we can prove that the alleviated scheme fulfills the following relaxed uniformity requirement, which fortunately is enough for designing secure chameleon signature schemes (see [KR00]).

**DEFINITION 4.15 (RELAXED UNIFORMITY FOR TRAPDOOR HASH FAMILIES)** *Let  $TH = (\text{KeyGen}, \text{H})$  be a trapdoor hash family and let  $\mathcal{A}$  be an adversary playing the following game:*

**GAME.UNI:**

Step 1.  $(\text{hk}, \text{tk}) \leftarrow \text{KeyGen}(1^k)$

Step 2.  $(m_0, m_1, st) \leftarrow \mathcal{A}(\text{hk})$

Step 3.  $b \leftarrow \{0, 1\}$

Step 4.  $r \leftarrow \mathcal{R}$

Step 5.  $h \leftarrow \text{hash}_{\text{hk}}(m_b; r)$

Step 6.  $\tilde{b} \leftarrow \mathcal{A}(h, st)$

The value  $st$  is an internal state information. We define the advantage of the adversary  $\mathcal{A}$  as  $\epsilon_{\mathcal{A}} = \left| \Pr[\tilde{b} = b] - \frac{1}{2} \right|$ . Moreover, we define  $\epsilon$  as  $\max_{\mathcal{A}}(\epsilon_{\mathcal{A}})$ , where the maximum is taken over all adversaries modeled as probabilistic polynomial time Turing machines. TK is said to meet the requirements of relaxed uniformity if  $\epsilon$  is negligible in the security parameter  $k$ .

LEMMA 4.4 *Relaxed Uniformity holds for the alleviated scheme under the Decisional Composite Residuosity Assumption.*

PROOF If  $s \in \{0, 1\}^{l+1}$ ,  $r \in \mathbb{Z}_n^\times$  are chosen uniformly at random, then for any two messages  $m_0, m_1$ , the distributions of  $\text{hash}_{\text{hk}}(m_0; r, s)$  and  $\text{hash}_{\text{hk}}(m_1; r, s)$  are computationally indistinguishable. This is an immediate consequence of the semantic security of the encryption scheme described in Section 4.3 (Theorem 4.6).  $\square$

REMARK 4.10 In terminology of commitment schemes, the above construction is a perfectly hiding and computationally binding trapdoor commitment scheme. For the alleviated scheme both hiding and binding holds computationally.

#### 4.6.4 Comparison

In this section, we compare previously known trapdoor hash families with our new constructions. In Table 4.5, we focus on schemes that are intended to be used in Shamir-Tauman on-line/off-line signature schemes. Here, the most important property is efficient weak altering. Furthermore, schemes allowing strong altering are preferable because they lead to more powerful conversions. In the last column, we note if the secret trapdoor has to be known to the sender. This is no problem in the suggested applications (on-line/off-line signatures, on-line/off-line signcryption), but for more general applications those schemes might be improper.

Scheme	Assumption	strong	hash	weak alt.	user needs tk
[BK90]	Discrete log	NO	$\approx 1$ exp.	$\approx 1$ mult.	NO
[KR00]	FACT	<b>YES</b>	$\approx  m _2$ mult.	$\approx 5$ mult.	NO
[ST01]	FACT	NO	1 exp.	<b>1 add. + bit shift</b>	YES
[BCP03]	FACT	NO	$\approx 1$ exp.	$\approx 1$ mult.	YES
1. proposed	FACT	<b>YES</b>	1 exp.	<b>1 add. + bit shift</b>	YES

Table 4.5: Comparison of trapdoor hash families suitable for [ST01]

In Table 4.6, we compare different constructions for on-line/off-line trapdoor hash families. Here, it is notable that the need for the user to know the secret trapdoor excludes the application chameleon signatures, therefore we designate the alleviated version of our second scheme (where the randomness is sampled from  $\{0, 1\}^{l+1}$ ).

Scheme	Assumption	strong	hash	user needs tk
[CGHGN01]	RSA( $n, n$ )	YES	$\approx 1$ exp.	<b>NO</b>
[BCP03]	<b>FACT</b>	NO	$\approx 1$ exp.	YES
2. proposed	<b>FACT</b>	YES	$\approx 1$ exp.	<b>NO</b>

Table 4.6: Comparison of on-line/off-line trapdoor hash families

In both tables, the assumption factoring refers to integers of  $p^2q$ -type for the proposed schemes and to RSA moduli, resp. Blum integers [KR00], else.

## 4.7 Conclusion

In this chapter, we introduced a new simple trapdoor one-way function and two associated trapdoor permutations based on the hardness of factoring. As provably secure trapdoor permutations are so rare and nevertheless of outstanding importance in public key cryptography, the development of new candidates is a fundamental issue on its own. Moreover, to constitute the claim that our proposed trapdoor function is not only of theoretical interest, we constructed several applications.

First, we pointed out that combining Paillier's homomorphic encryption scheme with the Okamoto-Uchiyama modulus  $n = p^2q$  yields a new homomorphic encryption scheme that is one-way under the  $p^2q$  Factorization Assumption.

Then, we introduced a novel CCA-secure hybrid encryption scheme to exploit that one of our proposed trapdoor permutations can be used to encrypt arbitrary strings like keys. To do so, we made use of the recently published Tag-KEM/DEM framework for hybrid encryption. We were able to show that even our proposed ad-hoc construction is more efficient than the members of the well-known EPOC family, which are based on the same intractability assumption as our proposal.

Next, we revisited some FSS schemes based on factorization related assumptions. We introduced a new FSS scheme based on the  $p^2q$  Factorization Assumption and provided a complete security proof in the bundling homomorphism framework. We also relaxed this framework to obtain more powerful factorization-based FSS schemes. These considerations may be of independent interest. Our new bundling homomorphism is more elemental and artless than the previous factoring bundling homomorphism from the vHPP scheme [PP97]. In addition, our entire construction compares favorably with the vHPP scheme which is based on the hardness of factoring a special kind of Blum integers.

As a further application, we constructed two new factorization-based trapdoor hash families. The first one provides as fast weak altering as Shamir-Tauman's state-of-the-art solution, but additionally allows strong altering, too, yet leading to more powerful generic constructions of on-line/off-line signature schemes. The second one is the first factorization-based trapdoor hash family that enables the construction of on-line/off-line chameleon signatures (a previous construction exists based on the hardness of inverting  $\text{RSA}(n, n)$ ).

## **Part II**

# **Efficient Implementation**



## Chapter 5

# Algorithms for Scalar Multiplication in Abelian Groups

In modern cryptosystems, one of the most important basic operation is the *scalar multiplication*  $dg = \underbrace{g + \dots + g}_d$ , where  $g$  is an element of an additively written Abelian

group  $(G, +)$  and  $d$  is an integer<sup>1</sup>. Note that this chapter is not intended to present a comprehensive treatment of the subject, instead, we provide the background for our own research presented in the subsequent chapters. Neither we describe scalar multiplication methods tailored to special groups (such as the use of normal bases in the case of  $\mathbb{F}_{p^n}$  [vzG91, Sti90], or efficiently computable endomorphisms on suitable elliptic curves [GLV01]), nor we deal with the subject of performing several multiplications of same element  $g$  with various scalars (in this case, methods are known that involve a relatively large amount of precomputation depending on  $g$ , but the final multiplications are sped up significantly [BGMW93, LL94]).

This chapter is organized as follows: in Section 5.1, we develop a common structure for performing scalar multiplication, whereas concrete realizations are described in Section 5.2 for generic groups and in Section 5.3 for groups where inversion is easy.

### 5.1 The General Scheme of Scalar Multiplication

A non-zero positive integer  $d$  is uniquely represented by a binary string:

$$d = \sum_{i=0}^{n-1} d_i 2^i, \quad d_i \in \{0, 1\}, \quad d_{n-1} = 1.$$

Here, we also write  $d = d_{n-1}|d_{n-2}|\dots|d_1|d_0$ .

The most common method for performing scalar multiplication is the double-and-add algorithm, which computes  $dg$  according to the bits  $d_i$  (therefore it is often called binary

---

<sup>1</sup>In this thesis, we only consider additively written groups because we will focus on elliptic curves in the subsequent chapters. In multiplicatively written groups, the corresponding operation is denoted as exponentiation.

method). In general, we distinguish two main concepts, namely scanning the bits left-to-right (l-t-r, see Algorithm 5), or right-to-left (r-t-l, see Algorithm 6), respectively.

---

**Algorithm 5:** Binary method, left-to-right
 

---

**Input:** a group element  $g$ , a non-zero  $n$ -bit scalar  $d = d_{n-1}| \dots |d_1|d_0$

**Output:** scalar multiplication  $dg$

```

1  $h \leftarrow g$ ;
2 for  $i = n - 2$  down to  $0$  do
3    $h \leftarrow 2h$ ;
4   if  $d_i = 1$  then  $h \leftarrow h + g$ 
5 return  $h$ 

```

---



---

**Algorithm 6:** Binary method, right-to-left
 

---

**Input:** a group element  $g$ , a non-zero  $n$ -bit scalar  $d = d_{n-1}| \dots |d_1|d_0$

**Output:** scalar multiplication  $dg$

```

1  $h_1 \leftarrow g$ ;  $h_2 \leftarrow 0_G$ ;
2 for  $i = 0$  to  $n - 1$  do
3   if  $d_i = 1$  then  $h_2 \leftarrow h_1 + h_2$ ;
4    $h_1 \leftarrow 2h_1$ 
5 return  $h_2$ 

```

---

The efficiency of this procedure may be enhanced if precomputation is allowed. In this case, we consider more general representations of the scalar, where each non-zero bit  $d_i$  is not restricted to be 1, but is an element of a suitable digit set  $\mathcal{T}$  of integers. We call  $d = \sum_{i=0}^{n-1} d_i 2^i$  a  $\mathcal{T}$ -representation, if  $d_{n-1} \neq 0$  and  $d_i \in \mathcal{T} \cup \{0\}$  holds for each  $0 \leq i < n$ . For brevity, we also write  $d = [d_{n-1}, \dots, d_1, d_0]_2$  in this case. In general,  $\mathcal{T}$ -representations lose the property of uniqueness, as the example  $[1, 0, 1]_2 = 5 = [2, 1]_2$  demonstrates.

The left-to-right double-and-add algorithm is easily adjusted to work with a  $\mathcal{T}$ -representation of the scalar<sup>2</sup>, namely addition of  $g$  in step 4 is replaced with addition of the precomputed elements  $d_i g$ , where  $d_i \in \mathcal{T}$  is the appropriate digit of  $d$ .

Hence, a general scheme for scalar multiplication can be described as follows: in the *precomputation stage* the points  $tg$  for  $t \in \mathcal{T}$  are precomputed and stored, in the *recoding stage* the scalar is rewritten to a  $\mathcal{T}$ -representation, and in the *evaluation stage* eventually the scalar multiplication is performed. Algorithm 7 illustrates this framework with a left-to-right evaluation phase.

It is now easy to see that the important features of a  $\mathcal{T}$ -representation are the number of non-zero digits and the cardinality of  $\mathcal{T}$ , because these quantities determine the required time and memory consumption for computing  $dg$ , respectively. The more precomputed elements are available, the less non-zero elements will remain in the recoded scalar, hence the faster the evaluation stage can be performed. Consequently, we are confronted with an optimization problem to determine a representation class with best trade-off between

---

<sup>2</sup>There is also a right-to-left method suitable for  $\mathcal{T}$ -representations [Knu81, Yao76]. For several reasons which are described in more detail in Section 6.4.1, the left-to-right evaluation stage is preferred in our intended applications.

---

**Algorithm 7:**  $\mathcal{T}$ -supported scalar multiplication, l-t-r evaluation

---

**Input:** a group element  $g$ , a non-zero scalar  $d$ , a description of a digit set  $\mathcal{T}$ **Output:** scalar multiplication  $dg$ **Precomputation:**

- 1 Compute  $tg$  for each  $t \in \mathcal{T}$

**Recoding:**

- 2 Obtain a  $\mathcal{T}$ -representation  $[d_{n-1}, \dots, d_1, d_0]_2$  of  $d$

**Evaluation:**

- 3  $h \leftarrow g$ ;
  - 4 **for**  $i = n - 2$  **down to**  $0$  **do**
  - 5      $h \leftarrow 2h$ ;
  - 6     **if**  $d_i \neq 0$  **then**  $h \leftarrow h + d_i g$
  - 7 **return**  $h$
- 

high non-zero density and low memory consumption. In this context it is expedient to remark that there is a simple but notable speed-up possible for groups where inversion (*i. e.*, subtraction) is easy (*e. g.*, point groups of elliptic curves or class-groups of imaginary quadratic number fields). In this case, if both  $t$  and  $-t$  are elements of  $\mathcal{T}$ , it is sufficient to precompute  $tg$ , as  $-tg$  can be generated on the fly. Consequently, for each precomputed point we get one additional precomputed point virtually for free. Here, if  $\mathcal{T}$  contains negative digits as well, we speak of *signed* representations of the scalar.

A further improvement is possible if both recoding and evaluation are of the same type (l-t-r resp. r-t-l). In this case, it is possible to merge both stages, which makes it needless to store the entire recoded scalar as an intermediate result in working memory. For l-t-r processing, this issue is dealt with in Chapter 6.

The characterizations of precomputation and recoding phase are left vague in Algorithm 7, because these phases cannot be generalized easily. The precomputation phase usually exploits an appropriate addition chain depending on the actual set  $\mathcal{T}$ . For example, in most of the subsequently considered algorithms,  $\mathcal{T}$  equals the set of all odd numbers below some bound  $2k + 1$ . In this case, the elements  $\{g, 3g, \dots, (2k + 1)g\}$  can be computed with only  $k + 1$  additions as follows: First, compute  $2g = g + g$ . Then, for  $i = 1, \dots, k$  compute  $(2i + 1)g = (2i - 1)g + 2g$ . Note that both summands in the last sum are already known, thus the sequence  $1, 2, 3, 5, \dots, 2k + 1$  forms an addition chain.

Now it remains to describe the recoding phase in more detail. As noted above, groups where inversion is easy require a special treatment, because we can exploit signed representations in this case. Therefore we distinguish these groups from the general case. As the main emphasis of the subsequent Chapters 6 and 7 are signed methods, we examine these techniques in more detail.

## 5.2 Known Solutions: The Generic Case

As noted in the preceding section, the scalar multiplication algorithms within our scope mainly differ in the recoding stage only. Consequently, we focus on various techniques

for generating  $\mathcal{T}$ -representations in the following. In this section, we restrict ourselves to the case where  $\mathcal{T}$  contains solely positive digits. The signed case will be considered in Section 5.3.

The most established techniques for generating  $\mathcal{T}$ -representations are *window methods*. Loosely speaking, in the window method with width  $w$  successively  $w$  consecutive bits of the binary scalar are scanned and, if necessary, replaced by a table-entry according to  $\mathcal{T}$ . Again, left-to-right and right-to-left processing of bits is possible. We distinguish fixed window methods like the  $2^w$ -ary method, where the window segmentation of the binary string is predetermined and the more advanced sliding window methods, where zero runs are skipped. More precisely, in a *sliding window scheme*, a fixed width window “slides” over the input bit stream, and, whenever the leftmost, resp. rightmost (referring to l-t-r, resp. r-t-l scanning) window bit is non-zero, the appropriate replacement takes place. Instead of recapitulating the algorithms (see [Knu81, MvOV96, Gor98] for this purpose), we illustrate the differences by the following examples for  $w = 3$  (the non-underbraced bits are transferred unchanged):

$$\begin{aligned}
&2^3\text{-ary, left-to-right: } \underbrace{101}_{005} \underbrace{110}_{006} \underbrace{001}_{001} \underbrace{011}_{003} \underbrace{110}_{006} \underbrace{000}_{000} \underbrace{110}_{006} \underbrace{001}_{001} \underbrace{011}_{003} \underbrace{011}_{003} 0 \\
&2^3\text{-ary, right-to-left: } 1 \underbrace{011}_{003} \underbrace{100}_{004} \underbrace{010}_{002} \underbrace{111}_{007} \underbrace{100}_{004} \underbrace{001}_{001} \underbrace{100}_{004} \underbrace{010}_{002} \underbrace{110}_{006} \underbrace{110}_{006} \\
&\text{width-3 sliding window, left-to-right: } \underbrace{101}_{005} \underbrace{110}_{030} 00 \underbrace{101}_{005} \underbrace{111}_{007} 0000 \underbrace{110}_{030} 00 \underbrace{101}_{005} \underbrace{101}_{005} 10 \\
&\text{width-3 sliding window, right-to-left: } 10 \underbrace{111}_{007} 000 \underbrace{101}_{005} \underbrace{111}_{007} 000 \underbrace{011}_{003} 0 \underbrace{001}_{001} \underbrace{011}_{003} \underbrace{011}_{003} 0
\end{aligned}$$

Note that the sliding window method halves the number of precomputed elements, because only odd multiples are required. In contrast, the  $2^w$ -ary method generates a fixed pattern  $0^{w-1} \star 0^{w-1} \star \dots$ , where  $\star \in \{0, 1, 2, \dots, 2^{w-1}\}$ , which is desirable in some applications<sup>3</sup>. It is also possible to do without the even precomputed elements in the  $2^w$ -ary method (*modified*  $2^w$ -ary method, see [BSS99], p. 65), but the sliding window method utilizes the precomputation table in a more effective manner<sup>4</sup>. Consequently, in the unsigned case (*i. e.*,  $\mathcal{T}$  consists solely of positive integers), sliding window techniques are the method of choice.

### 5.3 Known Solutions: The Case Where Inversion Is Easy

As said before, in groups where inversion is fast it is meaningful to consider *signed*  $\mathcal{T}$ -representations of the scalar. The precomputation effort reduces to the computation of  $|t|g$  for all  $t \in \mathcal{T}$ , because  $-tg$  can be derived from  $tg$  online. Unfortunately, the generation of signed  $\mathcal{T}$ -representations is less immediate than in the unsigned case. Overall, there are two strategies. The first one is to construct a  $\{-1, +1\}$  representation of  $d$  (also called a *signed binary representation*) and to apply window methods afterwards. Here, the most common

<sup>3</sup>Uniformity of the recoded scalar provides resistance against simple side channel attacks. There are also methods to avoid the case  $\star = 0$ , and consequently to achieve a perfect uniform output [Möl01].

<sup>4</sup>Roughly speaking, this effect can be qualified as using one bit larger sliding windows compared to fixed windows with the same amount of precomputation. Indeed, the sliding window method with width  $w$  achieves an asymptotic non-zero density of  $1/(w+1)$  compared to  $(1/w)(1-1/2^w)$  for the  $2^w$ -ary method [KYLH94].

signed binary representation is NAF (*non-adjacent-form*, referring to the property that no two consecutive bits are non-zero) [Rei60,INS00].

The second strategy is to generalize the NAF recoding for  $w > 2$  in order to obtain  $w$ NAF [Sol00,BSS99] (here, the non-adjacency property states that among any  $w$  adjacent bits, at most one is non-zero). According to [BSS99], this strategy is asymptotically the optimal one for  $w > 3$ . We sketch both methods in the following. In Section 6.3.3, alternative solutions for generating signed representations are analyzed.

### 5.3.1 NAF And Windowed NAF

As stated above, the signed binary representation NAF has been introduced by G. W. Reitwiesner in 1960 [Rei60], and it is characterized by the property that among any two adjacent digits, at most one is non-zero.

NAF can be obtained from the binary representation by applying the conversion  $\star|1|1 \mapsto \star + 1|0|\bar{1}$  repeatedly, where  $\bar{1}$  denotes  $-1$  and  $\star$  stands for any binary digit. However, the carry-over  $+1$  occurring in the first digit forces the recoding to be performed from the least significant bit, *i. e.*, right-to-left. Procedure NAF describes an easily implementable NAF generation.

---

#### Procedure NAF( $d$ )

---

**Input:** a non-zero scalar  $d = d_{n-1}| \dots |d_1|d_0$

**Output:** the NAF representation  $(\nu_n, \dots, \nu_0)$  of  $d$ , where  $\nu_i \in \{0, \pm 1\}$

```

1  $i \leftarrow 0; k \leftarrow d;$ 
2 while  $k > 0$  do
3   if  $k$  is odd then
4      $\nu_i \leftarrow k \bmod 4;$ 
5      $k \leftarrow k - \nu_i;$ 
6   else  $\nu_i \leftarrow 0;$ 
7    $k \leftarrow k/2; i \leftarrow i + 1;$ 
8 return  $(\nu_n, \dots, \nu_0)$ 
```

---

In line 4, we write " $b \bmod m$ " for the signed residue of  $b$  modulo  $m$ , that is:  $b \bmod m$  is the unique integer  $a$  in  $\mathbb{Z}$  which verifies  $(a \equiv b \pmod{m})$  and  $-\lfloor \frac{m-1}{2} \rfloor \leq a \leq \lfloor \frac{m}{2} \rfloor$ .

Reitwiesner was able to show that—ignoring leading zeros—each integer has a unique NAF representation. For this reason, some authors call NAF a canonical signed binary representation [EcKK94]. In addition, as shown among others by Jedwab and Mitchell [JM89], NAF representation provides the minimal Hamming weight (by *Hamming weight* we understand the number of non-zero digits). The non-zero density of NAF is only  $1/3$ , and consequently, we save about  $1/6$  of additions required for evaluating  $dg$  if  $d$  is represented as NAF instead of binary (the number of doublings is not affected). Therefore, the NAF representation of the scalar is the optimal choice if signed methods are meaningful and no precomputation is considered.

However, the situation is less clear if extra memory is available and precomputation is admitted. In this case, general signed representations outperform signed binary representations. As noted above, one strategy for constructing signed representations is the

application of window methods to signed binary representations. But as signed binary representation is redundant, the question arises which representation is the best for this purpose. Indeed, this is assumed to be an open problem by De Win *et al.* [WMPW98]. As NAF provides the minimal Hamming weight among all signed binary representations, it seems to be a good candidate [WMPW98, Ava02]. Algorithm 9 illustrates the resultant scalar multiplication algorithm.

---

**Algorithm 9:** Sliding window applied to NAF [WMPW98]

---

**Input:** a group element  $g$ , a non-zero  $n$ -bit scalar  $d$ , a width  $w$

**Output:** scalar multiplication  $dg$

**Precomputation:**

```

1  $g_1 \leftarrow g; g_2 \leftarrow 2g_1;$ 
2 for  $j$  from 1 to  $\left(\frac{2^{w+(-1)^{w+1}}}{3} - 1\right)$  do  $g_{2j+1} \leftarrow g_{2j-1} + g_2;$ 
   /* $g_{2j+1}$  contains  $(2j+1)g$  for each  $j$  in  $\{0, \dots, \frac{2^{w+(-1)^{w+1}}}{3} - 1\}$  */

```

**Recoding and Evaluation:**

```

3  $(\nu_n, \dots, \nu_0) \leftarrow \text{NAF}(d);$ 
4  $h \leftarrow 0_G; i \leftarrow n - 1;$ 
5 while  $i \geq 0$  do
6   if  $\nu_i = 0$  then  $h \leftarrow 2h; i \leftarrow i - 1;$ 
7   else
8      $s \leftarrow \max(i - w + 1, 0);$ 
9     /* $s$  is the index of the last digit of the new window */
10    Let  $t$  be the smallest integer such that  $t \geq s$  and  $\nu_t \neq 0;$ 
11    for  $k$  from 1 to  $i - t + 1$  do  $h \leftarrow 2h;$ 
12    /*(computation of  $2^{i-t+1}h$ ) */
13    if  $[\nu_i, \nu_{i-1}, \dots, \nu_t]_2 > 0$  then
14       $h \leftarrow h + g_{[\nu_i, \nu_{i-1}, \dots, \nu_t]_2};$ 
15    if  $[\nu_i, \nu_{i-1}, \dots, \nu_t]_2 < 0$  then
16       $h \leftarrow h - g_{|[\nu_i, \nu_{i-1}, \dots, \nu_t]_2|};$ 
17    for  $k$  from 1 to  $t - s$  do  $h \leftarrow 2h;$ 
18    /*(computation of  $2^{t-s}h$ ) */
19     $i \leftarrow s - 1;$ 

```

17 **return**  $A$

---

Note that although the sliding window method in the evaluation stage can be performed left-to-right, it is necessary to proceed right-to-left to obtain the NAF representation of  $d$ . Thus, this whole representation has to be stored as an intermediate result. In Chapter 6, we will develop a signed representation not suffering from this drawback.

### 5.3.2 $w$ NAF

The natural generalization of NAF is  $w$ NAF which is defined as follows<sup>5</sup>.

---

<sup>5</sup>Alternative generalizations of Reitwiesner's NAF recoding idea can be found in [Pro00, Avi61].

DEFINITION 5.1 (*wNAF*) *A sequence of signed digits is called wNAF iff the following three properties hold:*

1. *The most significant non-zero bit is positive.*
2. *Among any  $w$  consecutive digits, at most one is non-zero.*
3. *Each non-zero digit is odd and less than  $2^{w-1}$  in absolute value.*

It seems that *wNAF* has first been described by Miyaji, Ono and Cohen [MOC97]. The algorithm proposed in [MOC97] is rather involved, and Solinas gave a more elegant description [Sol00]. Instead of applying window methods to signed binary representations, *wNAF* is constructed directly from unsigned binary using a generalization of Procedure NAF. Note that the original NAF is the same as *wNAF* for  $w = 2$ .

---

**Procedure** NAF( $d, w$ )

---

**Input:** a non-zero scalar  $d = d_{n-1}| \dots |d_1|d_0$ , a width  $w > 1$

**Output:** the *wNAF* representation  $(\nu_w[n], \dots, \nu_w[0])$  of  $d$ , where

$$\nu_w[i] \in \{0, \pm 1, \pm 3, \dots, \pm 2^{w-1} - 1\}$$

```

1  $i \leftarrow 0; k \leftarrow d;$ 
2 while  $k > 0$  do
3   if  $k$  is odd then
4      $\nu_w[i] \leftarrow k \bmod 2^w;$ 
5      $k \leftarrow k - \nu_w[i];$ 
6   else  $\nu_w[i] \leftarrow 0;$ 
7    $k \leftarrow k/2; i \leftarrow i + 1;$ 
8 return  $(\nu_w[n], \dots, \nu_w[0])$ 

```

---

Recently, Muir and Stinson proved the well-known fact that the *wNAF* of an integer is at most one digit longer than its binary representation [MS06]. In addition, Muir and Stinson were able to show that *wNAF* provides the minimal number of non-zero digits of all  $\{\pm 1, \pm 3, \dots, \pm 2^{w-1} - 1\}$ -representations (an alternative proof has been given by Avanzi [Ava04]). This fact has been assumed to be true for years, but a formal proof was only known for the case  $w = 2$ . Note that this property does not imply superiority of *wNAF* compared with sliding window on NAF (see Section 5.3.1), because the latter requires a larger digit set for the same width  $w$ . This results in a higher precomputation effort, but it turns out that the evaluation stage is slightly faster than for *wNAF*. Indeed, to compare both methods rigorously, one has to develop explicit formulae for the number of group operations necessary in precomputation and evaluation stage, moreover the precomputation effort has to be adjusted for a fair comparison. This has been done by Blake, Seroussi and Smart [BSS99] with the conclusion that *wNAF* is preferable for  $w > 3$ , though the margin of difference is slim. The crucial point for determining the estimated number of group additions in a *wNAF* supported evaluation stage is the observation that the asymptotic non-zero density of *wNAF* equals  $1/(w + 1)$ . This result has been independently proved by Miyaji, Ono and Cohen [MOC97] on the one hand and by Solinas [Sol00] on the other hand.

As the class *wNAF* is of high relevance for the subsequent chapters, we summarize its most important features in

THEOREM 5.1 *wNAF as defined in Definition 5.1 has the following properties:*

1. *Each positive integer possesses a wNAF representation.*
2. *The length of the wNAF representation exceeds the binary representation by at most one digit.*
3. *wNAF provides the minimal number of non-zero digits of all  $\mathcal{T}$ -representations for  $\mathcal{T} = \{\pm 1, \pm 3, \dots, \pm 2^{w-1} - 1\}$ .*
4. *The asymptotic non-zero density of wNAF equals  $1/(w + 1)$ .*

For the sake of completeness, Algorithm 11 illustrates the scalar multiplication with wNAF. Again, the recoding stage is done right-to-left, therefore it is not possible to merge recoding and evaluation.

---

**Algorithm 11:** Scalar multiplication with width- $w$  NAF [BSS99]

---

**Input:** a point  $g$ , a non-zero  $n$ -bit scalar  $d$ , a width  $w$

**Output:** the point  $dg$

**Precomputation:**

- 1  $g_1 \leftarrow g; g_2 \leftarrow 2g_1;$
- 2 **for**  $j$  *from* 1 *to*  $2^{w-2} - 1$  **do**  $g_{2j+1} \leftarrow g_{2j-1} + g_2;$
- $/*g_{2j+1}$  contains  $(2j + 1)g$  for each  $j$  in  $\{0, \dots, 2^{w-2} - 1\}$  \*/

**Recoding:**

- 3  $(\nu_w[n], \dots, \nu_w[0]) \leftarrow \text{NAF}(d, w);$

**Evaluation:**

- 4 Let  $c$  be the largest integer with  $\nu_w[c] \neq 0;$
- 5 **if**  $\nu_w[c] > 0$  **then**  $h \leftarrow g_{\nu_w[c]};$
- 6 **if**  $\nu_w[c] < 0$  **then**  $h \leftarrow -g_{|\nu_w[c]|};$
- 7 **for**  $i$  *from*  $c - 1$  *down to* 0 **do**
- 8      $h \leftarrow 2h;$
- 9     **if**  $\nu_w[i] > 0$  **then**  $h \leftarrow h + g_{\nu_w[i]};$
- 10    **if**  $\nu_w[i] < 0$  **then**  $h \leftarrow h - g_{|\nu_w[i]|};$

11 **return**  $h$

---

## Chapter 6

# MOF—A New Canonical Signed Binary Representation With Applications to Elliptic Curve Cryptography

The most common method for computing scalar multiplication of random elements in Abelian groups are sliding window schemes, which enhance the efficiency of the binary method at the expense of some precomputation (see Chapter 5). In groups where inversion is easy, signed representations of the exponent are meaningful because they decrease the amount of required precomputation. The asymptotic best signed method is  $w$ NAF, because it minimizes the precomputation effort whilst its non-zero density is optimal. Unfortunately,  $w$ NAF can be computed only from the least significant bit, *i. e.*, right-to-left. In connection with memory constraint devices, however, left-to-right recoding schemes are by far more valuable.

In this chapter, we define the MOF (*Mutually Opposite Form*), a new canonical representation of signed binary strings, which can be computed in any order. Therefore we obtain the first left-to-right signed recoding scheme for general width  $w$  by applying the width  $w$  sliding window conversion on MOF left-to-right. Moreover, the analogue right-to-left conversion on MOF yields  $w$ NAF, which indicates that the new class is the natural left-to-right analogue to the useful  $w$ NAF. Indeed, the new class inherits the outstanding properties of  $w$ NAF, namely the required precomputation and the achieved non-zero density are exactly the same.

### 6.1 Introduction

As the ubiquitous computing devices are penetrating our daily life, the importance of memory constraint devices (*e. g.*, smart cards) in cryptography is increasing. Note that in connection with these devices, the most popular cryptosystems are based on elliptic curve point groups [Kob87, Mil85], because elliptic curve cryptosystems (ECC) provide high security with moderate key-lengths. Hence, ECC seems to be the future standard, especially for hand-held devices which have scarce resources. The most important operation

in ECC is scalar multiplication (*i. e.*, the multiplication of an integer with a point of the curve) which has been discussed in the preceding chapter. Recall that scalar multiplication of a random point is split into three phases: first, a fairly small amount of *precomputation* depending on the particular point and the set  $\mathcal{T}$  determined by the selected window method has to be performed, then – in the *recoding* phase – the scalar is rewritten to a  $\mathcal{T}$ -representation, and finally – in the *evaluation* stage – the multiplication is done.

### 6.1.1 New Motivation for Memory-saving Scalar Multiplication Algorithms

Modern smart cards are equipped with a few Kbytes RAM only and most of them are reserved for OS and stack. Thus, cryptographic algorithms and especially elliptic curve scalar multiplication should be optimized in terms of memory. In particular, we are reluctant to consume memory except the necessary precomputation related to  $\mathcal{T}$ .

As noted in Chapter 5, scalar recoding may be performed from the least significant bit (right-to-left) and from the most significant bit (left-to-right), respectively. For the purpose of ECC on memory constraint devices, however, we prefer left-to-right to right-to-left recoding methods. The reason is as follows: In the case of elliptic curve scalar multiplication, the left-to-right evaluation stage is the natural choice (see Section 6.4.1 for details). If the scalar recoding is done right-to-left, it is necessary to finalize the recoding and to store the entire recoded string in working memory before starting the left-to-right evaluation stage. In other words, we require additional  $n$ -bit RAM for the right-to-left recoding, where  $n$  is the bit size of the scalar.

On the contrary, if a left-to-right recoding technique is available, the recoding and evaluation stage may be interleaved to obtain an efficient scalar multiplication on the fly, without storing the recoded scalar at all. Therefore it is an important task to construct a left-to-right recoding scheme, even if the size of  $\mathcal{T}$  and the non-zero density are not improved.

### 6.1.2 Left-to-Right and Carry-free Generation Of Signed Representations

In Chapter 5, we have seen that in the unsigned case applying a standard sliding window scheme on the binary representation is the method of choice<sup>1</sup>. However, a nice property of elliptic curves is that inversion is computed virtually for free. In this case, it is meaningful to consider signed representations of the scalar. In Chapter 5 we recalled some standard methods for generating signed representations, but unfortunately, due to the occurrence of carry-overs, the recoding is restricted to be done right-to-left in all cases. Consequently, all scalar multiplication strategies based on signed  $\mathcal{T}$ -representations require  $\mathcal{O}(n)$  bits of additional working memory to store the recoded scalar as an intermediate result. Solely in the case of  $w = 2$ , Joye and Yen proposed a left-to-right signed binary recoding algorithm [JY00]. But it has been an unsolved problem to generate a left-to-right recoding algorithm for a larger set  $\mathcal{T}$ .

From Section 5.3.2, we know that the asymptotic non-zero density of  $w$ NAF is the

<sup>1</sup>Recall that we consider the case where neither the scalar nor the group element are known in advance, and in addition we assume that some memory is available to store precomputed data.

same as for the unsigned sliding window method on binary, namely  $1/(w+1)$ . Moreover, if the latter is applied right-to-left, its output also fulfills the non-adjacency property (among any  $w$  consecutive digits, at most one is non-zero). Therefore,  $w$ NAF can be seen as the natural signed analogue of the standard sliding window method on binary, and we guess that there could be a carry-free generation method for  $w$ NAF. Here, the term carry-free refers to an algorithm that transforms the input string in situ, *i. e.*, in each step only the knowledge of a fixed number of consecutive input bits is necessary.

In this chapter, we solve both problems as follows: (1) we define a new canonical representation class of signed binary. We call it MOF (Mutually Opposite Form) and prove that each integer can be uniquely represented as MOF. But the outstanding property of MOF is that it can be efficiently developed from a binary string right-to-left or left-to-right, likewise. Consequently, analogue to the unsigned case, sliding window methods may be applied to receive left-to-right and right-to-left recoding schemes for general width  $w$ . Surprisingly, applying the right-to-left width  $w$  sliding window method on MOF yields  $w$ NAF. The observation that in the unsigned case right-to-left sliding window yields an unsigned string with non-adjacency property stresses the analogy between unsigned binary and signed MOF. Therefore we achieve a *carry-free*  $w$ NAF generation, a benefit of its own.

(2) Our major aim is the development of a left-to-right recoding algorithm, and this is achieved straightforwardly by applying the width  $w$  sliding window method left-to-right on MOF. We call the so-defined class  $w$ MOF and prove that each integer can be uniquely represented as a  $w$ MOF and that the asymptotic non-zero density of  $w$ MOF equals  $1/(w+1)$ , which is the same as for  $w$ NAF. Therefore the classes  $w$ NAF and  $w$ MOF may be seen as dual to each other. In general our proposed algorithm asymptotically requires additional  $\mathcal{O}(w)$  bits of RAM for storing intermediate results, which is independent from the bit size  $n$  and dramatically reduces the required space compared with previous methods. Consequently, due to its left-to-right nature, the new scheme is by far more convenient with respect to memory consumption than previous schemes. Interestingly, a straightforward proof shows that for  $w = 2$  the proposed method produces the same output as the Joye-Yen recoding, but 2MOF is more efficient in terms of counting the number of basic operations.

We finish this chapter with an explicit algorithm for on-the-fly elliptic curve scalar multiplication with  $w$ MOF, proving that the proposed schemes are indeed useful for practical purposes.

### 6.1.3 Related Works

There is a large amount of literature available on efficient exponentiation techniques<sup>2</sup> (see, *e. g.*, the textbooks [Knu81, MvOV96] and the survey paper [Gor98]). In the following, we only cite selected results that are important within the scope of this chapter. It was first pointed out by Morain and Olivos that NAF can be used to speed up elliptic curve scalar multiplication [MO90]. De Win *et al.* suggested the application of a sliding window scheme on NAF (NAF+SW) to obtain a signed representation with smaller Hamming weight [WMPW98]. A different approach to construct lower-weighted signed representations is the generalization of NAF recoding for  $w > 2$  in order to obtain  $w$ NAF [Sol00, BSS99]. Both recoding methods are of right-to-left type and have been

<sup>2</sup>Elliptic curve scalar multiplication is a special instance of exponentiation in Abelian groups.

reviewed in Section 5.3. Before the submission of the paper [OSSST04a], which is the foundation of this chapter, the only known left-to-right recoding technique was restricted to the signed *binary* case [JY00]. But interestingly, several authors independently dealt with the topic of left-to-right signed digit representations at the same time [Ava04,MS05,HKPR05]. In [HGPT05], Heuberger, Grabner, Prodinger and Thuswaldner introduce the so-called *alternating-greedy-expansion* of integers, a left-to-right computable signed binary representation that turns out to be the same as MOF in a different description (implemented as a greedy algorithm). In a subsequent paper, Heuberger, Katti, Prodinger and Ruan propose the application of window methods to the alternating-greedy-expansion, therefore they rediscovered *w*MOF [HKPR05]. Avanzi also constructed a left-to-right algorithm that produces *w*MOF [Ava04], but his description is by far less intuitive than ours. A further difference is that Avanzi proves optimality<sup>3</sup>—*i. e.*, minimal Hamming weight among all representations with  $\mathcal{T} \subset \{1, 2, \dots, 2^{w-1}\}$ —whereas we deduce the asymptotic non-zero density of *w*MOF. Both results are of high value: the minimality fact shows that *w*MOF is indeed the method of choice for the intended applications, whilst knowledge of the non-zero density is required when analyzing the efficiency of *w*MOF supported scalar multiplication (the amount of non-vanishing digits determines the number of point additions to be performed during the evaluation stage). A different approach has been published by Muir and Stinson [MS05], namely they define a class of left-to-right computable minimal weight representations, and they present a *probabilistic* algorithm for computing members of this class. It is notable that *w*MOF is a possible outcome of this algorithm ([Mui04], Section 3.6).

## 6.2 MOF: New Canonical Representation for Signed Binary Strings

In this section, we present a new signed representation of integers, called the mutually opposite form (MOF). We assume that MOF can serve as a new *canonical* signed binary representation dual to unsigned binary for the following reasons:

1. For each non-negative integer, a MOF representation exists and is unique when leading zeros are ignored.
2. The asymptotic non-zero density of MOF is 1/2 as for unsigned binary.
3. MOF and unsigned binary lead to analogue results when window methods are applied to them.

Some authors call NAF a canonical representation [EcKK94], but this view is solely based on the uniqueness of NAF encoding. We assume that the arguments in favor of MOF are more convincing.

We start with a formal definition of MOF:

**DEFINITION 6.1 (MOF)** *The  $n$ -bit mutually opposite form (MOF) is an  $n$ -bit signed binary string that satisfies the following properties:*

---

<sup>3</sup>This result is shown for *w*MOF as well as for *w*NAF.

1. The signs of adjacent non-zero bits (without considering zero bits) are opposite.
2. The most non-zero bit and the least non-zero bit are 1 and  $\bar{1}$ , respectively, unless all bits are zero.

Some zero bits are inserted between non-zero bits that have a mutually opposite sign. An example of MOF is 0100 $\bar{1}$ 01000 $\bar{1}$ 001 $\bar{1}$ 0. For simplicity, we only deal with non-negative integers in the following. If the most significant bit of MOF is not restricted to be positive, negative integers can be encoded too. All of the following results easily carry over to this case.

An important observation is that each positive integer can be uniquely represented by MOF. Indeed, the following theorem is proved in [OSSST04a].

**THEOREM 6.1** *Let  $n$  be a positive integer. There are exactly  $2^n$  pair-wise different signed binary strings of length  $n + 1$  that meet Definition 6.1. Thus, there is the bijective map between elements of  $(n + 1)$ -bit MOF and  $n$ -bit binary strings.*

From this theorem, any  $n$ -bit binary string can be uniquely represented by  $(n + 1)$ -bit MOF. We obviously have the following corollary about the non-zero density of MOF.

**PROPOSITION 6.1** *The average non-zero density of  $n$ -bit MOF is  $1/2$  for  $n \rightarrow \infty$ .*

### 6.2.1 Converting Binary String to MOF

We show a simple and flexible conversion from  $n$ -bit binary string to  $(n + 1)$ -bit MOF. The crucial point is the following observation: The  $n$ -bit binary string  $d$  can be converted to a signed binary string by computing  $\mu = 2d \ominus d$ , where  $\ominus$  stands for a bitwise subtraction. Indeed, we convert  $d$  as follows:

$$\begin{array}{rcccccccc} & 2d = d_{n-1} & | & d_{n-2} & | & \dots & | & d_{i-1} & | & \dots & | & d_1 & | & d_0 & | \\ \ominus & d = & | & d_{n-1} & | & \dots & | & d_i & | & \dots & | & d_2 & | & d_1 & | & d_0 \\ \hline & \mu = d_{n-1} & | & d_{n-2} - d_{n-1} & | & \dots & | & d_{i-1} - d_i & | & \dots & | & d_1 - d_2 & | & d_0 - d_1 & | & -d_0. \end{array}$$

Here the  $i$ -th signed bit of  $\mu$  is denoted by  $\mu_i$ , namely  $\mu_i = d_{i-1} - d_i$  for  $i = 1, \dots, n - 1$  and  $\mu_n = d_{n-1}, \mu_0 = -d_0$ . In [OSSST04a] it is proved that the signed representation  $\mu$  is MOF.

**PROPOSITION 6.2** *The operation  $\mu = 2d \ominus d$  converts a binary string  $d$  to its MOF  $\mu$ .*

Procedure MOF provides an explicit left-to-right conversion from Binary to MOF.

---

#### Procedure MOF( $d$ )

---

**Input:** a non-zero integer  $d = d_{n-1}|d_{n-2}|\dots|d_1|d_0$

**Output:** MOF representation  $(\mu_n, \dots, \mu_1, \mu_0)$  of  $d$ , where  $\mu_i \in \{0, \pm 1\}$

**1**  $d_{-1} \leftarrow 0; d_n \leftarrow 0;$

**2** **for**  $i = n$  **down to**  $0$  **do**  $\mu_i \leftarrow d_{i-1} - d_i;$

**3** **return**  $(\mu_n, \mu_{n-1}, \dots, \mu_1, \mu_0)$

---

The assignments in the first line capture the cases  $\mu_n = d_{n-1}$  and  $\mu_0 = -d_0$ . In order to generate the  $i$ -th bit  $\mu_i$ , Procedure MOF stores just two consecutive bits  $d_{i-1}$  and  $d_i$ . This algorithm converts a binary string to MOF from the most significant bit in an efficient way. Note that it is possible to convert a binary string to MOF right-to-left as well. Thus, the MOF representation is highly flexible.

REMARK 6.1 Interestingly, the MOF representation of an integer  $d$  equals the recoding performed by the classical Booth algorithm for binary multiplication [Boo51]. The classical Booth algorithm successively scans two consecutive bits of the multiplier  $A$  (right-to-left). Depending on these bits, one of the following operations is performed:

No operation,	if $(a_i, a_{i-1}) \in \{(0, 0), (1, 1)\}$ ,
Subtract multiplicand $B$ from the partial product,	if $(a_i, a_{i-1}) = (1, 0)$ ,
Add multiplicand $B$ to the partial product,	if $(a_i, a_{i-1}) = (0, 1)$ ,

where  $a_{-1}$  is defined as 0. Of course, the design goal of this algorithm was to speed up multiplication when there are consecutive ones in the multiplier  $A$ , and to provide a multiplication method that works for signed and unsigned numbers as well. To the best of our knowledge, this representation never served as a fundament of theoretical treatment of signed binary strings.

### 6.3 Window Methods on MOF

In this section, we show how to decrease the non-zero density of MOF by applying window methods on it. First we consider the right-to-left width  $w$  sliding window method which surprisingly yields the familiar  $w$ NAF. In contrast to previously known generation methods, the new one is carry-free, *i. e.*, in each step the knowledge of at most  $w + 1$  consecutive input bits is sufficient.

Then we define the dual new class  $w$ MOF as the result of the analogue left-to-right width  $w$  sliding window method on MOF. This conversion leads to the first left-to-right signed recoding scheme for general width  $w$ .

#### 6.3.1 Right-to-Left Case: $w$ NAF

There are several algorithms for generating  $w$ NAF, for example see [BSS99, MOC97, Sol00], but each method needs carry-overs. Note that in the worst case all remaining bits are affected by the carry, therefore the previously known  $w$ NAF algorithms can not be considered as local methods. By inspecting the  $w$ NAF generation procedure given in Section 5.3.2 closely, we observe that this generation can be seen as the natural signed analogue to the right-to-left sliding window method on (unsigned) binary (here, mod instead of mods is computed). Indeed, the latter method produces a representation that fulfills the non-adjacency requirement. Consequently, we conjecture that there might be a signed binary representation that produces  $w$ NAF when handled with sliding window conversions. In this section, we point out that MOF serves for this purpose.

In order to describe the proposed scheme, we need the conversion table for width  $w$ . First, we define the conversions for MOF windows of length  $l$ , where the first and the last bit is non-zero:

$$\begin{array}{ccc}
\underbrace{0|\dots|0|2^{l-2}+1}_{l} \leftarrow \left\{ \begin{array}{l} 1|\bar{1}|0|\dots|0|0|1 \\ 1|\bar{1}|0|\dots|0|1|\bar{1} \end{array} \right. & \underbrace{0|\dots|0|2^{l-2}+3}_{l} \leftarrow \left\{ \begin{array}{l} 1|\bar{1}|0|\dots|0|1|0|\bar{1} \\ 1|\bar{1}|0|\dots|0|1|\bar{1}|1 \end{array} \right. & \dots \\
\dots & \underbrace{0|\dots|0|2^{l-1}-3}_{l} \leftarrow \left\{ \begin{array}{l} 1|0|\dots|0|\bar{1}|1|\bar{1} \\ 1|0|\dots|0|\bar{1}|0|1 \end{array} \right. & \underbrace{0|\dots|0|2^{l-1}-1}_{l} \leftarrow \left\{ \begin{array}{l} 1|0|\dots|0|0|\bar{1} \\ 1|0|\dots|0|\bar{1}|1 \end{array} \right.
\end{array}$$

In addition, we have analogue conversions with all signs changed. To generate the complete table for width  $w$ , we have to consider all conversions of length  $l = 2, 3, \dots, w$ . If  $l < w$  holds, the window is filled with leading zeros.

**EXAMPLE 6.1 (CONVERSION TABLE FOR 4NAF)** In the case of  $w = 4$ , we use the following table  $\text{Table}_{4\overleftarrow{SW}}$  for the right-to-left sliding window method:

$$\begin{array}{cccc}
0001 \leftarrow \left\{ \begin{array}{l} 0001 \\ 001\bar{1} \end{array} \right. & 0003 \leftarrow \left\{ \begin{array}{l} 010\bar{1} \\ 01\bar{1}1 \end{array} \right. & 0005 \leftarrow \left\{ \begin{array}{l} 1\bar{1}01 \\ 0\bar{1}1\bar{1} \end{array} \right. & 0007 \leftarrow \left\{ \begin{array}{l} 100\bar{1} \\ 10\bar{1}1 \end{array} \right. \\
000\bar{1} \leftarrow \left\{ \begin{array}{l} 000\bar{1} \\ 00\bar{1}1 \end{array} \right. & 000\bar{3} \leftarrow \left\{ \begin{array}{l} 0\bar{1}01 \\ 0\bar{1}1\bar{1} \end{array} \right. & 000\bar{5} \leftarrow \left\{ \begin{array}{l} \bar{1}10\bar{1} \\ 01\bar{1}1 \end{array} \right. & 000\bar{7} \leftarrow \left\{ \begin{array}{l} \bar{1}001 \\ \bar{1}01\bar{1} \end{array} \right.
\end{array}$$

In an analogue way  $\text{Table}_{w\overleftarrow{SW}}$  is defined for general  $w$ . Based on this table, Algorithm 13 provides a simple carry-free  $w$ NAF generation.

---

**Algorithm 13:** Carry-free right-to-left Generation from Binary to  $w$ NAF

---

**Input:** a width  $w$ , a non-zero  $n$ -bit binary string  $d = d_{n-1}|d_{n-2}|\dots|d_1|d_0$

**Output:**  $w$ NAF representation  $\nu_w = (\nu_w[n], \dots, \nu_w[0])$  of  $d$ , where

$$\nu_w[i] \in \{0, \pm 1, \pm 3, \dots, \pm 2^{w-1} - 1\}$$

1  $d_{n+w-2} \leftarrow 0; d_{n+w-3} \leftarrow 0; \dots; d_n \leftarrow 0; d_{-1} \leftarrow 0; i \leftarrow 0;$

2 **while**  $i \leq n$  **do**

3     **if**  $d_{i-1} = d_i$  **then**

4          $\nu_w[i] \leftarrow 0; i \leftarrow i + 1$

5     **else**

6         /\*The MOF window begins with a non-zero digit righthand

\*/

6          $(\nu_w[i+w-1], \nu_w[i+w-2], \dots, \nu_w[i]) \leftarrow$

7          $\text{Table}_{w\overleftarrow{SW}}(d_{i+w-2} - d_{i+w-1}, d_{i+w-3} - d_{i+w-2}, \dots, d_{i-1} - d_i);$

7          $i \leftarrow i + w$

8 **return**  $(\nu_w[n], \dots, \nu_w[0])$

---

Obviously, the output of Algorithm 13 meets the notations of Definition 5.1, therefore it is  $w$ NAF. If we knew that Definition 5.1 provides a *unique* representation, we could deduce that Algorithm 13 outputs the same as Procedure  $w$ NAF from Section 5.3.2. This is true, although we could not find a proof in literature. For the sake of completeness, we prove the following theorem via exploiting the uniqueness of MOF representation.

**THEOREM 6.2** *Every non-negative integer  $d$  has a representation as  $w$ NAF, which is unique except for the number of leading zeros.*

PROOF We show (ignoring leading zeros in the following) that Definition 5.1 leads to a unique representation of positive integers as follows:

1. We define a conversion  $\text{MOF} \longrightarrow w\text{NAF}$ .
2. We define a conversion  $w\text{NAF} \longrightarrow \text{MOF}$ .
3. We show that these two conversion are inverse to each other.

Consequently, there is a bijection between  $w\text{NAF}$  and  $\text{MOF}$ . As there is also a bijection between  $\text{MOF}$  and Binary, this proves the uniqueness of  $w\text{NAF}$ .

**ad (1):**  $\text{MOF} \longrightarrow w\text{NAF}$  is defined by performing the sliding window method with width  $w$  from the least significant bit (*i. e.*, right-to-left) on  $\text{MOF}$  as described in Section 6.3.1.

**ad (2):**  $w\text{NAF} \longrightarrow \text{MOF}$  scans the bits right-to-left using a window with width  $w$ , until the rightmost window digit  $d$  is non-zero. All scanned zeros are taken to the converted string as usual. If  $d$  is non-zero, due to the properties of  $w\text{NAF}$  the window content is of the shape  $\underbrace{0 \dots 0}_{w-1} d, d \in \{\pm 1, \pm 3, \dots, \pm 2^{w-1}\}$ . To find the correct replacement, we distinguish two cases:

**Case 1** The most significant non-zero bit of the already converted string equals -1:

We build the length- $w$ -MOF corresponding to  $|d|$  (padded with leading zeros, if necessary). In the case of  $d < 0$ , we change all signs of this length- $w$ -MOF and take this string. Otherwise, we force the last bit of the length- $w$ -MOF to be 1 by replacing its last 2 bits as follows:  $1\bar{1} \mapsto 01, 0\bar{1} \mapsto \bar{1}1$ .

**Case 2** The most significant non-zero bit of the already converted string equals 1 or no non-zero bit has been converted at all:

We build the length- $w$ -MOF corresponding to  $|d|$  (padded with leading zeros, if necessary). In the case of  $d > 0$ , we take this string. Otherwise, we change all signs of this length- $w$ -MOF and we force the last bit of this string to be  $\bar{1}$  by replacing its last 2 bits as follows:  $01 \mapsto 1\bar{1}, \bar{1}1 \mapsto 0\bar{1}$ .

This case differentiation ensures that the converted string possesses the MOF properties (particularly the alternating signs of the non-zero bits).

EXAMPLE 6.2 In the case of  $w = 3$ , we use the following table for the right-to-left conversion

$$\begin{array}{l} 001 \mapsto \begin{cases} 001 & \text{Case1} \\ 01\bar{1} & \text{Case2} \end{cases} \quad 00\bar{1} \mapsto \begin{cases} 0\bar{1}1 & \text{Case1} \\ 00\bar{1} & \text{Case2} \end{cases} \\ 003 \mapsto \begin{cases} 1\bar{1}1 & \text{Case1} \\ 10\bar{1} & \text{Case2} \end{cases} \quad 00\bar{3} \mapsto \begin{cases} \bar{1}01 & \text{Case1} \\ \bar{1}\bar{1}\bar{1} & \text{Case2} \end{cases} \end{array}$$

**ad (3):** The two conversions are inverse to each other, because if we perform *w. l. o. g.*  $\text{MOF} \longrightarrow w\text{NAF}$  first and  $w\text{NAF} \longrightarrow \text{MOF}$  afterwards, the fragmentations of the strings are exactly the same and the tables are inverse to each other.  $\square$

### 6.3.2 Left-to-Right Case: $w$ MOF

In this section, we introduce our new proposed recoding scheme. The crucial observation is that as the generation Binary  $\rightarrow$  MOF can be performed left-to-right, the combination of this generation and left-to-right sliding window method leads to a complete signed left-to-right recoding scheme dual to  $w$ NAF.

In order to describe the proposed scheme, we need the conversion table for width  $w$ . The conversions for MOF windows of length  $l$ , such that the first and the last bit is non-zero, are defined in exactly the same way as in the right-to-left case (see the table in Section 6.3.1 and reflect the assignments). To generate the complete table for width  $w$ , we have to consider all conversions of length  $l = 2, 3, \dots, w$  as before. The only difference is that if  $l < w$  holds, the window is filled with *closing* zeros instead of leading ones.

**EXAMPLE 6.3 (CONVERSION TABLE FOR 4MOF)** As an example, we construct the conversion table  $\text{Table}_{4\overrightarrow{SW}}$  for width 4:

$$\begin{array}{ccccc} 1000 \} \mapsto 1000 & 1\bar{1}00 \} \mapsto 0100 & \begin{array}{l} 1\bar{1}10 \\ 10\bar{1}0 \end{array} \} \mapsto 0030 & \begin{array}{l} 1\bar{1}01 \\ 1\bar{1}\bar{1}\bar{1} \end{array} \} \mapsto 0005 & \begin{array}{l} 100\bar{1} \\ 10\bar{1}\bar{1} \end{array} \} \mapsto 0007 \\ \bar{1}000 \} \mapsto \bar{1}000 & \bar{1}100 \} \mapsto 0\bar{1}00 & \begin{array}{l} \bar{1}\bar{1}\bar{1}0 \\ \bar{1}0\bar{1}0 \end{array} \} \mapsto 00\bar{3}0 & \begin{array}{l} \bar{1}\bar{1}0\bar{1} \\ \bar{1}\bar{1}\bar{1}\bar{1} \end{array} \} \mapsto 000\bar{5} & \begin{array}{l} \bar{1}00\bar{1} \\ \bar{1}0\bar{1}\bar{1} \end{array} \} \mapsto 000\bar{7} \end{array}$$

The table is complete due to the properties of MOF. Note that because of the equalities  $\star 1\bar{1} = \star 01$ ,  $\star \bar{1}1 = \star 0\bar{1}$  usually two different MOF-strings are converted to the same pattern. In an analogue way,  $\text{Table}_{w\overrightarrow{SW}}$  is defined for general width  $w$ . In this case, the digit set equals  $\mathcal{T} = \{\pm 1, \pm 3, \dots, \pm 2^{w-1} - 1\}$ , which is the same as for  $w$ NAF. Therefore, the scheme requires only  $2^{w-2}$  precomputed elements. Algorithm 14 makes use of this table to generate  $w$ MOF left-to-right.

In order to deepen the duality between  $w$ NAF and  $w$ MOF, we give a formal definition of  $w$ MOF and prove that it leads to a unique representation of non-negative integers.

**DEFINITION 6.2** *A sequence of signed digits is called  $w$ MOF iff the following three properties hold:*

1. *The most significant non-zero bit is positive.*
2. *All but the least significant non-zero digit  $x$  are adjoint by  $w-1$  zeros as follows:*
  - *in case of  $2^{k-1} < |x| < 2^k$  for an integer  $2 \leq k \leq w-1$  the pattern equals*

$$\underbrace{0 \dots 0}_k x \underbrace{0 \dots 0}_{w-k-1},$$
  - *in case of  $|x| = 1$  either the pattern equals  $x \underbrace{0 \dots 0}_{w-1}$  and the next lower non-zero digit has opposite sign from  $x$  or the pattern equals  $0x \underbrace{0 \dots 0}_{w-2}$  and the next lower non-zero digit has the same sign as  $x$ .*

*If  $x$  is the least significant non-zero digit, it is possible that the number of right-hand adjacent zeros is smaller than stated above. In addition it is not possible that the last non-zero digit is a 1 following any non-zero digit.*

3. Each non-zero digit is odd and less than  $2^{w-1}$  in absolute value.

This definition is directly related to the generation of  $w$ MOF. Note that the exceptional case corresponding to the least significant bit takes in account that the last window may be shorter than  $w$ .

---

**Algorithm 14:** Left-to-Right Generation from Binary to  $w$ MOF

---

**Input:** a width  $w$ , a non-zero  $n$ -bit binary string  $d = d_{n-1}|d_{n-2}|\dots|d_1|d_0$

**Output:**  $w$ MOF representation  $\delta_w = (\delta_w[n], \delta_w[n-1], \dots, \delta_w[1], \delta_w[0])$  of  $d$ , where  
 $\delta_w[i] \in \{0, \pm 1, \pm 3, \dots, \pm 2^{w-1} - 1\}$

```

1  $d_{-1} \leftarrow 0; d_n \leftarrow 0; i \leftarrow n;$ 
2 while  $i \geq w - 1$  do
3   if  $d_i = d_{i-1}$  then
4      $\delta_w[i] \leftarrow 0; i \leftarrow i - 1$ 
5   else
6     /*The MOF window begins with a non-zero digit lefthand */
7      $(\delta_w[i], \delta_w[i-1], \dots, \delta_w[i-w+1]) \leftarrow$ 
       $\text{Table}_{wSW}(d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{i-w} - d_{i-w+1});$ 
8      $i \leftarrow i - w$ 
9   if  $i \geq 0$  then
10     $(\delta_w[i], \delta_w[i-1], \dots, \delta_w[0]) \leftarrow \text{Table}_{i+1SW}(d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_0 - d_1, -d_0)$ 
11 return  $(\delta_w[n], \delta_w[n-1], \dots, \delta_w[1], \delta_w[0])$ 

```

---

Regarding the uniqueness and the non-zero density of  $w$ MOF, we have the following two theorems:

**THEOREM 6.3** *Every non-negative integer  $d$  has a representation as  $w$ MOF, which is unique except for the number of leading zeros.*

**PROOF** We proceed as follows:

1. We define a conversion  $\text{MOF} \longrightarrow w\text{MOF}$ .
2. We define a conversion  $w\text{MOF} \longrightarrow \text{MOF}$ .
3. We show that these two conversion are inverse to each other.

Consequently, there is a bijection between  $w$ MOF and MOF. As there is also a bijection between MOF and Binary, this proves the uniqueness of  $w$ MOF. This proof is similar to the preceding one ( $\text{MOF} \leftrightarrow w\text{NAF}$ ).

**ad (1):**  $\text{MOF} \longrightarrow w\text{MOF}$  is defined by performing the sliding window method with width  $w$  from the most significant bit (*i. e.*, left-to-right) on MOF.

**ad (2):**  $w\text{MOF} \longrightarrow \text{MOF}$  scans the bits right-to-left using a window with width  $w$ , until the window content equals one of the patterns described in Definition 6.2, property 2. All scanned zeros are taken to the converted string as usual. The replacements are performed as in the proof of Theorem 6.2 with the following exceptions: If the window



is asymptotically better than sliding window on NAF schemes if  $w > 3$  [BSS99]. In the context of memory constraint devices, a small digit set  $\mathcal{T}$  is even more valuable, because fewer precomputed elements have to be stored. But as none of the preceding methods is a left-to-right scheme, each one requires *additional* memory  $\mathcal{O}(n)$  to store the recoded string before starting the left-to-right evaluation of the scalar product. Note that in the context of sliding window on signed binary schemes like [KT92, WMPW98] the sliding window conversion may be performed left-to-right, but to obtain the signed binary representation we have to proceed right-to-left in either case.

In contrast,  $w$ MOF turns out as a complete left-to-right scheme. Consequently, there is no additional memory required for performing the scalar multiplication. Moreover, due to the properties of MOF, the digit set of  $w$ MOF is the same as for  $w$ NAF and therefore minimal.

Next, we compare the characterizing properties for the proposed schemes and some previous ones. In the second column, the value  $\#\mathcal{T}/2$  equals the number of elements that have to be precomputed and stored. In the last column, we describe the amount of memory (in bits) that is required additionally to this storage, *e. g.*, to construct the signed representation or to store the converted string in right-to-left schemes. As usual,  $n$  equals the bit-length of the scalar, and SW is an abbreviation for sliding window.

Scheme	$\#\mathcal{T}/2$	1/N.-z. Density	Additional Memory
$w$ NAF [Sol00, BSS99, MOC97]	$2^{w-2}$	$w + 1$	$\mathcal{O}(n)$
[KT92]	$2^{w-1} - 1$	$w + \frac{3}{2}$	$\mathcal{O}(n)$
NAF+SW as [WMPW98]	$\frac{1}{3}(2^w + (-1)^{w+1})$	$w + \frac{4}{3} - \frac{(-1)^w}{3 \cdot 2^{w-2}}$	$\mathcal{O}(n)$
$w$ MOF, Sec. 6.3.2	$2^{w-2}$	$w + 1$	$\mathcal{O}(w)$

Table 6.1: Comparison of Memory Requirement and Non-zero Density

## 6.4 Applications to Elliptic Curve Cryptography

In this section, we show how MOF representation can be exploited to develop memory saving algorithms for elliptic curve scalar multiplication, which is the most important operation in elliptic curve cryptography.

### 6.4.1 Elliptic Curve Scalar Multiplication

Let  $K = \mathbb{F}_p$  be a finite field, where  $p > 3$  is a prime. Let  $E$  be an elliptic curve over  $K$ . The elliptic curve  $E$  has an Abelian group structure with identity element  $\mathcal{O}$  called the point of infinity. The elliptic curve additions  $P_1 + P_2$  and  $2P$  are denoted by ECADD and ECDBL, respectively, where  $P_1, P_2, P \in E$ . ECADD and ECDBL are constructed with the arithmetic of the base field  $K$ . There are several coordinate systems that provide efficient algorithms for computing ECADD and ECDBL [CMO98], but in each coordinate system, the inversion of a point can be computed immediately. For example, in affine coordinates a point  $P$  is represented as  $P = (x, y)$ ,  $x, y \in \mathbb{F}_p$ , and  $-P$  is then obtained as  $-P = (x, -y)$ .

In Section 5.1, we have observed that scalar multiplication can be performed left-to-right and right-to-left, respectively. Though in general both methods provide the same

efficiency, the left-to-right method is preferable due to the following reasons:

1. The left-to-right binary method can be adjusted for general  $\mathcal{T}$ -representations of  $d$  like  $w$ NAF or  $w$ MOF in a more efficient way than the right-to-left method. Although there is a right-to-left analogue of Algorithm 7, this variant is less flexible as the “precomputation” depends on both the point *and the scalar* [Knu81, Yao76].
2. The ECADD step in the left-to-right method (*cf.* line 6 in Algorithm 7) has the fixed input  $tP$ ,  $t \in \mathcal{T}$ . Therefore it is possible to speed up these steps if  $tP$  is expressed in affine coordinates for each  $t \in \mathcal{T}$ , since some operations are negligible in this case. The approach of using different coordinate systems for different operations is known as *mixed coordinates method*, and the improvement for a 160-bit scalar multiplication is about 15% with NAF over right-to-left scheme in the Jacobian coordinates [CMO98].

#### 6.4.2 Explicit Implementation

Algorithm 15 puts the ideas of Section 6.3.2 into practice. It is notable that in contrast to previous solutions reviewed in Section 5.3 the recoding and evaluation phase can be merged.

## 6.5 Conclusion

It was an unsolved problem to generate a signed representation *left-to-right* for a general digit set  $\mathcal{T}$ . In this chapter, we presented a solution of this problem. The proposed scheme inherits the outstanding properties of  $w$ NAF, namely the set of pre-computed elements and the non-zero density are same as those of  $w$ NAF. In order to achieve a left-to-right scalar recoding, we defined a new canonical representation of signed binary strings, called the mutually opposite form (MOF). An  $n$ -bit integer can be uniquely represented by  $(n + 1)$ -bit MOF, and this representation can be constructed efficiently left-to-right. Then the proposed recoding is obtained by applying the width  $w$  (left-to-right) sliding window conversion to MOF. The proposed scheme is conceptually easy to understand and it is quite simple to implement. Moreover, if we apply the width  $w$  (right-to-left) sliding window conversion to MOF, we surprisingly obtain the classical  $w$ NAF. This is the first *carry-free* algorithm for generating  $w$ NAF. Therefore the proposed scheme has a lot of advantages and it promises to be a good alternative to  $w$ NAF. We believe that there will be many new applications of this algorithms for cryptography.

**Algorithm 15:** Scalar Multiplication with  $w$ MOF**Input:** a point  $P$ , a width  $w$ , a non-zero  $n$ -bit binary string  $d = d_{n-1}|d_{n-2}|\dots|d_1|d_0$ **Output:** the point  $dP$ **Precomputation:**

```

1  $P_1 \leftarrow P; P_2 \leftarrow \text{ECDBL}(P_1);$ 
2 for  $j$  from 1 to  $2^{w-2} - 1$  do  $P_{2j+1} \leftarrow \text{ECADD}(P_{2j-1}, P_2);$ 
   /*( $P_{2j+1}$  contains  $(2j + 1)P$  for each  $j$  in  $\{0, \dots, 2^{w-2} - 1\}$ ) */

```

**Recoding and Evaluation:**

```

3  $A \leftarrow \mathcal{O}; i \leftarrow n;$ 
4  $d_{-1} \leftarrow 0; d_n \leftarrow 0;$ 
5 while  $i \geq 0$  do
6   if  $d_{i-1} = d_i$  then  $A \leftarrow \text{ECDBL}(A); i \leftarrow i - 1;$ 
   /*(the MOF digit is zero) */
7   else
8      $s \leftarrow \max(i - w + 1, 0);$ 
     /* $s$  is the index of the last digit of the new window */
9     Let  $t$  be the smallest integer such that  $t \geq s$  and  $d_{t-1} \neq d_t$ ;
10    for  $k$  from 1 to  $i - t + 1$  do  $A \leftarrow \text{ECDBL}(A);$ 
     /*(computation of  $2^{i-t+1}A$ ) */
11    if  $[d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{t-1} - d_t]_2 > 0$  then
12       $A \leftarrow \text{ECADD}(A, P_{[d_{i-1}-d_i, d_{i-2}-d_{i-1}, \dots, d_{t-1}-d_t]_2});$ 
13    if  $[d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{t-1} - d_t]_2 < 0$  then
14       $A \leftarrow \text{ECADD}(A, -P_{|[d_{i-1}-d_i, d_{i-2}-d_{i-1}, \dots, d_{t-1}-d_t]_2|});$ 
15    for  $k$  from 1 to  $t - s$  do  $A \leftarrow \text{ECDBL}(A);$ 
     /*(computation of  $2^{t-s}A$ ) */
16     $i \leftarrow s - 1;$ 

```

17 **return**  $A$

## Chapter 7

# Fractional MOF—Elliptic Curve Scalar Multiplication With Flexible Memory Consumption

In the preceding chapter, we introduced  $w$ MOF, a left-to-right recoding technique perfectly suited for performing elliptic curve scalar multiplication. Though  $w$ MOF turned out to be advantageous compared to the previous methods sliding window on NAF (NAF+SW) and  $w$ NAF, it nevertheless shares a common drawback with these schemes: Only a small portion of numbers are possible sizes for the precomputation tables, and the gaps between consecutive table sizes grow exponentially in the width  $w$ . Therefore, in practice it is often necessary to waste memory, because there is no table fitting exactly the available storage. Only in the case of  $w$ NAF, there exists a variant that allows arbitrary table sizes, the so-called fractional  $w$ NAF (Frac- $w$ NAF).

In this chapter, we propose the fractional  $w$ MOF (Frac- $w$ MOF), which is a left-to-right analogue of Frac- $w$ NAF. We provide a comprehensive proof using Markov theory to determine the non-zero density of Frac- $w$ MOF. It turns out that Frac- $w$ MOF inherits the outstanding properties of Frac- $w$ NAF. However, because of its left-to-right nature, Frac- $w$ MOF is preferable as it reduces the memory consumption of the scalar multiplication.

Finally, we show that the properties of all discussed previous schemes can be achieved as special instances of the Frac- $w$ MOF method. To demonstrate the practicability of Frac- $w$ MOF, we develop an on-the-fly algorithm for computing elliptic curve scalar multiplication with a flexibly chosen amount of memory.

Since it has been recently proved that Frac- $w$ MOF additionally provides minimal Hamming weight among all representations using an appropriate digit set, we conclude that the Frac- $w$ MOF method is the optimal choice for performing elliptic curve scalar multiplication in memory-constraint devices.

### 7.1 Introduction

In Section 5.3 we reviewed NAF+SW and  $w$ NAF, two well-known representations suitable for performing elliptic curve scalar multiplication of random points. For devices with scarce

resources, however, these two as well as all further previous techniques turned out to be sub-optimal due the right-to-left/left-to-right mismatch of recoding and evaluation stage. To overcome this problem, we introduced  $w$ MOF—a left-to-right analogue of  $w$ NAF—in the preceding chapter. But unfortunately, a common drawback of all previous methods also applies to  $w$ MOF, namely the table sizes used to store precomputed elements are not arbitrary (for example, we have  $\#\{|x| : x \in \mathcal{T}\} = 2^{w-2}$  for  $w$ NAF and  $w$ MOF). Hence, the gaps between  $w$ -table and  $(w+1)$ -table grow exponentially fast with  $w$ . We achieve some additional values taking NAF+SW into account, but the actual problem persists. Consider the case that a certain amount of memory is available to store the precomputed table, but none of the possible tables fits exactly the obtainable memory. Subject to these limitations, we have to choose a smaller table and consequently waste memory. This is painful especially in memory-constraint devices like smart cards, for which  $w$ MOF is designated. To address this problem, Möller proposed fractional window methods [Möl02], in particular fractional  $w$ NAF (Frac- $w$ NAF) as a variant of  $w$ NAF for the signed case. The Frac- $w$ NAF allows a free choice of the table-size and is as efficient as  $w$ NAF.

In this chapter, we introduce the fractional  $w$ MOF (Frac- $w$ MOF) method, which is a left-to-right analogue of Frac- $w$ NAF. The proposed Frac- $w$ MOF is a highly flexible variant of  $w$ MOF as it can be set up with arbitrary table sizes. We will show that the average non-zero density of Frac- $w$ MOF for a given memory size is exactly equal to that of Frac- $w$ NAF, thus highlighting the duality of  $w$ NAF and  $w$ MOF once more.

From this result, we can achieve an efficient left-to-right recoding algorithm with flexible memory usage. As an application to elliptic curve cryptosystems, we present an on-the-fly scalar multiplication algorithm based on Frac- $w$ MOF recoding. The proposed scheme requires virtually no additional working memory<sup>1</sup>, and thus it is suitable for implementation on memory-constraint devices.

Finally, based on these results, we are able to give a detailed comparison of NAF+SW,  $w$ NAF, and  $w$ MOF with the fractional methods. From the construction of the latter it is obvious that  $w$ NAF and  $w$ MOF can be modeled with the fractional methods. But surprisingly, we point out that the same is true for NAF+SW which is generated in a completely different manner (see Section 5.3). Namely, we show that the properties of NAF+SW can be achieved as a special instance of Frac- $w$ MOF. Therefore, we conclude that the Frac- $w$ MOF is the best window method among these schemes.

Recently, independently from our work, Möller also analyzed a generalization of his fractional method to the left-to-right case. He achieved the same representation as Frac- $w$ MOF and was able to prove that this representation provides minimal Hamming weight among all signed-digit representations utilizing the digit set  $\{0, \pm 1, \pm 2, \dots, \pm m\}$  [Möl04]. This work goes perfectly with our results and confirms our conjecture that Frac- $w$ MOF is the best universally applicable method for performing base-2 elliptic curve scalar multiplication in limited-constrained devices.

### 7.1.1 Elementary Markov Theory

In this section, we review the most basic facts about Markov theory. For a comprehensive treatment, see [Häg02]. Markov theory is often used to analyze random processes over finite state spaces with the property that the probability of the current state depends only

<sup>1</sup>In case of Frac- $w$ NAF, the entire recoded scalar has to be buffered in working memory.

on the preceding one, but not on the earlier stages. Such a process  $X_0, X_1, \dots$  is called a *homogenous Markov chain*, if  $\Pr(X_{n+1}|X_n)$  is the same for all  $n$ . Homogeneous Markov chains can be represented by a *transition matrix* that contains the conditional probabilities of a stage given the preceding stage. Namely, if  $S = \{s_1, \dots, s_k\}$  is the state space, then the entries of the transition matrix  $P$  are given as  $P_{i,j} = \Pr(X_{n+1} = s_j | X_n = s_i)$ . A crucial property of homogenous Markov chains is that given the *initial distribution*, *i. e.*,  $\mu^{(0)} := (\Pr(X_0 = s_0), \Pr(X_0 = s_1), \dots, \Pr(X_0 = s_k))$ , the distribution at an arbitrary time  $n$  can be computed as  $\mu^{(n)} = (\Pr(X_n = s_0), \Pr(X_n = s_1), \dots, \Pr(X_n = s_k)) = \mu^{(0)} P^n$ .

A natural question now concerns the asymptotic behavior of the Markov chain, *i. e.*, the behavior of the sequence  $(\mu^{(n)})_{n \in \mathbb{N}}$  when  $n$  tends to infinity. To answer this question, we first consider two properties of Markov chains. The first one is *irreducibility*: Roughly speaking, a Markov chain is called irreducible, if each state  $s_i$  can be reached from each state  $s_j$ , that is  $\exists n \in \mathbb{N} : \Pr(X_n = s_i | X_0 = s_j) > 0$ . The second property is *aperiodicity*: A Markov chain is called aperiodic, if for all states  $s_i$  the greatest common divisor of the set of times the chain can return to  $s_i$  when starting from  $s_i$  equals one, that is  $\gcd(\{n \geq 1 \mid \Pr(X_n = s_i | X_0 = s_i)\}) = 1$ . It is possible that from a certain point on, the distribution of the Markov chain will stay invariant for all time, that is  $\exists k \forall i \geq k : \mu^{(i)} = \mu^{(k)}$ . In this case, the distribution  $\mu^{(k)}$  is called a *stationary distribution*<sup>2</sup>, and we have  $\mu^{(k)} P = \mu^{(k)}$  ( $\mu^{(k)}$  is an eigenvector of  $P$ ).

The main theorem of elementary Markov theory now states there exists a unique stationary distribution  $\pi$  for each irreducible and aperiodic Markov chain, and under an appropriate metric the sequence  $\mu^{(n)}$  tends to  $\pi$  with  $n \rightarrow \infty$ . Therefore we can analyze the asymptotic behavior of irreducible and aperiodic Markov chains simply by determining the eigenvector of its transition matrix.

## 7.2 Fractional Window Methods

The window methods described in the preceding section use precomputed tables whose sizes are not arbitrary, as shown in Fig. 7.1.

In particular, the gaps between two consecutive table sizes grow exponentially fast. This is a problem, especially in limited performance devices, such as smart cards: For these devices, we want to use all the available memory, in order to obtain a better trade-off between memory and speed. The fractional window methods fulfill this purpose.

### 7.2.1 Fractional $w$ NAF

The fractional  $w$ NAF (Frac- $w$ NAF) [Möl02] is a generalization of the  $w$ NAF method which permits the use of precomputed tables of any sizes. We denote with  $q$  the number of elements of the intended precomputed table. There is a unique  $w_0$  such that  $2^{w_0-2} \leq q < 2^{w_0-1}$  ( $w_0 = \lfloor \log_2(q) \rfloor + 2$ ). If we were using the standard  $w$ NAF algorithm, then the window size could be at most  $w_0$ , and the precomputed table would contain  $2^{w_0-2}$  elements. We now call  $r$  the size of the memory space which would be wasted ( $r = q - 2^{w_0-2}$ ) and we introduce the notation  $w_1$  for  $w_1 = r/2^{w_0-2}$ . Note that  $0 \leq r < 2^{w_0-2}$  and that  $0 \leq w_1 < 1$ . The elements in the precomputed table are  $\{1P, 3P, 5P, \dots, (2q-1)P\} = \{1P, 3P, 5P, \dots,$

<sup>2</sup>Note that this does not mean  $\forall i \geq k : X_i = X_k$ .

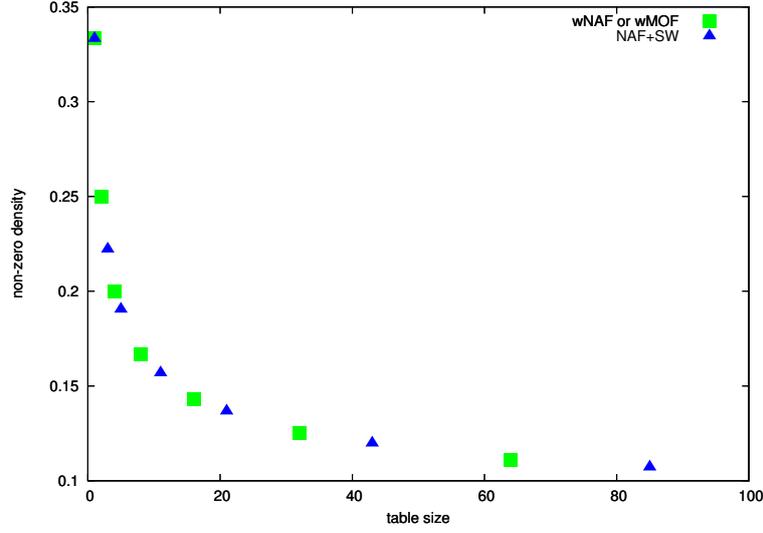


Figure 7.1: Concrete table sizes and corresponding non-zero densities

$(2^{w_0-1}-1)P, (2^{w_0-1}+1)P, \dots, (2^{w_0-1}+w_12^{w_0-1}-1)P$ . The precomputed table hence has  $q = (1+w_1)2^{w_0-2}$  elements. These parameters define *Frac- $w$ NAF* with width  $w = w_0 + w_1$ .

The generation of *Frac- $w$ NAF* of width  $w = w_0 + w_1$  in Procedure *Frac- $w$ NAF* from [Möl02] derives directly from the  $(w_0+1)$ NAF generation (see the Procedure given in Section 5.3), in which we replace the non precomputed elements by their  $w_0$ NAF analogue (lines 5 and 6 of Procedure *Frac- $w$ NAF*).

---

**Procedure *Frac- $w$ NAF*( $d, w$ )**


---

**Input:** a fractional width  $w = w_0 + w_1$  where  $w_0 = \lfloor w \rfloor$  and  $w_1 = r/2^{w_0-2}$  for an  $r$  between 0 and  $2^{w_0-2} - 1$ , a non-zero  $n$ -bit binary string

$$d = d_{n-1}|d_{n-2}| \dots |d_1|d_0$$

**Output:** the *Frac- $w$ NAF* representation  $(\nu_w[n], \nu_w[n-1], \dots, \nu_w[0])$  of  $d$ , where

$$\nu_w[i] \in \{0, \pm 1, \pm 3, \dots, \pm 2^{q-1} - 1\}, q = (1 + w_1)2^{w_0-2}$$

```

1  $i \leftarrow 0$ ;
2 while  $d > 0$  do
3   if  $d$  is odd then
4      $\nu_{w_0+1}[i] \leftarrow d \bmod 2^{w_0+1}$ ;
5     if  $|\nu_{w_0+1}[i]| > ((1 + w_1)2^{w_0-1} - 1)$  then  $\nu_w[i] \leftarrow d \bmod 2^{w_0}$ ;
6     else  $\nu_w[i] \leftarrow \nu_{w_0+1}[i]$ ;
7      $d \leftarrow d - \nu_w[i]$ ;
8   else  $\nu_w[i] \leftarrow 0$ ;
9    $d \leftarrow d/2$ ;  $i \leftarrow i + 1$ ;
10 return  $(\nu_w[n], \nu_w[n-1], \dots, \nu_w[0])$ 

```

---

We obtain an algorithm for scalar multiplication with *Frac- $w$ NAF* by replacing the re-coding stage of Algorithm 11 with the above *Frac- $w$ NAF* generation. In addition, the precomputation phase has to be adjusted as follows: instead of  $j = 0, \dots, 2^{w-2} - 1$  the loop passes through  $j = 0, \dots, 2^{w_0-2}(1 + w_1) - 1$  (such that the largest table entry is

$$(2j + 1)P = (2^{w_0-2} + w_1 2^{w_0-2} - 1)P).$$

In [Möl02], Möller states that the asymptotic non-zero density of Frac- $w$ NAF with a valid fractional window size  $w$  equals  $1/(w + 1)$ . This claim has been proved rigorously by Schmidt-Samoa *et al.* [SSST06].

### 7.2.2 Fractional $w$ MOF

In this section, we apply the highly flexible fractional window method to MOF.

As the precomputation is the same for  $w$ NAF and  $w$ MOF (see Section 6.3), we can argue in a similar way as in the preceding section. In particular, let  $r$  be the number of points that should be precomputed in addition to the table corresponding to width  $w_0$  ( $0 \leq r < 2^{w_0-2}$ ). Then we define the fractional width as  $w = w_0 + w_1$ , where  $w_1 = r/2^{w_0-2}$ . To evaluate the scalar multiplication, we proceed like  $(w_0 + 1)$ MOF if possible, and like  $w_0$ MOF, else. Algorithm 17 illustrates this concept.

In the first phase, the table entries are precomputed. There is memory available for  $2^{w_0-2} + r$  points, therefore we precompute  $P_1, P_3, \dots, P_{2^{w_0-1}-1}$  ( $2^{w_0-2}$  points) and  $P_{2^{w_0-1}+1}, P_{2^{w_0-1}+3}, \dots, P_{2^{w_0-1}+2r-1}$  ( $r$  points). With  $w_1 = r/2^{w_0-2}$  we obtain the bound for the loop variable  $j$ .

The second phase merges recoding and evaluation. Note that the  $i$ th MOF-digit  $\mu_i$  of  $d$  is computed as  $d_{i-1} - d_i$  and that the MOF representation is one bit longer than the binary representation. The assignments in line 4 are necessary to compute the MOF-digits  $\mu_0$  and  $\mu_n$ , respectively. The evaluation is performed from left-to-right, *i. e.*, from the most significant bit. Whenever  $d_{i-1} = d_i$  holds, the MOF digit  $\mu_i$  is zero and only an ECDBL is computed (line 6). Otherwise, a window of length  $w_0 + 1$  is processed. In line 8, the index  $s$  of the rightmost window digit is assigned. As we do not use the full precomputed table for width  $w_0 + 1$ , it may happen that the window content corresponds to a point that is not available. First we note that this is only possible if  $d_{s-1} \neq d_s$ , *i. e.*,  $\mu_s \neq 0$  holds: Let  $\mu_s = 0$  and let  $t > s$  be minimal with respect to  $d_{t-1} \neq d_t$ , *i. e.*,  $\mu_t \neq 0$ . Then the appropriate precomputed point corresponding to the actual window is computed as  $|\underbrace{[\mu_i, \mu_{i-1}, \dots, \mu_t]_2}_{w_0} \leq [1, 0, \dots, 0, \bar{1}]_2 = 2^{w_0-1} - 1$  following the MOF-property. Thus, the

precomputed table is always sufficient. In case of  $d_{s-1} \neq d_s$ , the precomputed table is too small if and only if  $|\underbrace{[\mu_i, \mu_{i-1}, \dots, \mu_s]_2}_{w_0} \geq 2^{w_0-1}(1 + w_1)$  holds. Consequently, the window has to be reduced in this case (line 10). In line 11, the index  $t$  of the window's rightmost non-zero MOF digit is assigned. The remaining part of the algorithm does not differ from the usual scalar multiplication with  $(w_0 + 1)$ MOF resp.  $w_0$ MOF (see Algorithm 15).

To finish this section, we prove the following theorem.

**THEOREM 7.1** *Let  $w$  be a valid fractional window size, *i. e.*,  $w = w_0 + w_1$  where  $w_0 = \lfloor w \rfloor$  and  $w_1 = r/2^{w_0-2}$  for an  $r$  between 0 and  $2^{w_0-2} - 1$ . The fractional  $w$ MOF scheme applied to a scalar  $d$  requires  $(1 + w_1)2^{w_0-2}$  precomputed elements and achieves a representation with a non-zero density of  $\frac{1}{w_0+w_1+1} = \frac{1}{w+1}$  when the bit length of  $d$  tends to infinity.*

**PROOF** From the considerations above, it is obvious that the number of precomputed points equals  $(1 + w_1)2^{w_0-2}$ .

We model the recoding process of Frac- $w$ MOF as a Markov process that on input of an infinitely long sequence of uniformly and independently distributed bits  $d_0, d_1, \dots$  outputs

**Algorithm 17:** Scalar Multiplication with Frac- $w$ MOF

**Input:** a point  $P$ , a fractional width  $w = w_0 + w_1$ , where  $w_0 = \lfloor w \rfloor$  and  $w_1 = r/2^{w_0-2}$  for an  $r$  between 0 and  $2^{w_0-2} - 1$ , a non-zero  $n$ -bit binary string  $d = d_{n-1}|d_{n-2}| \dots |d_1|d_0$ ,

**Output:** the point  $dP$

**Precomputation:**

1  $P_1 \leftarrow P$ ;  $P_2 \leftarrow \text{ECDBL}(P_1)$ ;  
 2 **for**  $j$  from 1 to  $2^{w_0-2}(1 + w_1) - 1$  **do**  $P_{2j+1} \leftarrow \text{ECADD}(P_{2j-1}, P_2)$ ;  
 /\*( $P_{2j+1}$  contains  $(2j + 1)P$  for each  $j$  in  $\{0, \dots, 2^{w_0-2}(1 + w_1) - 1\}$ ) \*/

**Recoding and Evaluation:**

3  $A \leftarrow \mathcal{O}$ ;  $i \leftarrow n$ ;  
 4  $d_{-1} \leftarrow 0$ ;  $d_n \leftarrow 0$ ;  
 5 **while**  $i \geq 0$  **do**  
 6     **if**  $d_{i-1} = d_i$  **then**  $A \leftarrow \text{ECDBL}(A)$ ;  $i \leftarrow i - 1$ ;  
    /\*(the MOF digit is zero) \*/  
 7     **else**  
 8          $s \leftarrow \max(i - w_0, 0)$ ;  
        /\* $s$  is the index of the last digit of the new window \*/  
 9         **if**  $d_{s-1} \neq d_s$  **and**  $|(d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{s-1} - d_s)_2| \geq 2^{w_0-1}(1 + w_1)$   
        **then**  
            /\*the window has to be reduced \*/  
             $s = s + 1$ ;  
 10         Let  $t$  be the smallest integer such that  $t \geq s$  and  $d_{t-1} \neq d_t$ ;  
 11         **for**  $k$  from 1 to  $i - t + 1$  **do**  $A \leftarrow \text{ECDBL}(A)$ ;  
        /\*(computation of  $2^{i-t+1}A$ ) \*/  
 12         **if**  $(d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{t-1} - d_t)_2 > 0$  **then**  
 13              $A \leftarrow \text{ECADD}(A, P_{|d_{i-1}-d_i, d_{i-2}-d_{i-1}, \dots, d_{t-1}-d_t|_2})$ ;  
 14         **if**  $(d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{t-1} - d_t)_2 < 0$  **then**  
 15              $A \leftarrow \text{ECADD}(A, -P_{|d_{i-1}-d_i, d_{i-2}-d_{i-1}, \dots, d_{t-1}-d_t|_2})$ ;  
 16         **for**  $k$  from 1 to  $t - s$  **do**  $A \leftarrow \text{ECDBL}(A)$ ;  
        /\*(computation of  $2^{t-s}A$ ) \*/  
 17          $i \leftarrow s - 1$ ;  
 18     

19 **return**  $A$

a series of blocks of the following shapes: a single zero bit, a block of  $w_0 + 1$  bits, or a block of  $w_0$  bits. Let the current bit be  $d_i$  and let  $s$  be defined like in Algorithm 17, *i. e.*, if  $d_i \neq d_{i-1}$  then  $s = i - w_0$  is the index of the rightmost digit in the current window. We distinguish three states:

- State  $b_1$ :  
The output is a single zero bit. This is the case iff  $d_{i-1} = d_i$  holds.
- State  $b_2$ :  
The output equals  $\underbrace{0^{i-t+1} \star 0^{t-i+w_0}}_{w_0+1}$ , where  $\star$  is a non-zero digit and  $t \geq s$  is minimal with respect to  $d_{t-1} \neq d_t$ . We define the following property:

$$d_{s-1} = d_s \text{ or } |[d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{s-1} - d_s]_2| \leq 2^{w_0-1}(1 + w_1) - 1 \quad (7.1)$$

Then state  $b_2$  occurs iff  $d_{i-1} \neq d_i$  holds and property (7.1) is fulfilled.

- State  $b_3$ :  
The output equals  $\underbrace{0^{i-t+1} \star 0^{t-i+w_0-1}}_{w_0}$ , where  $\star$  is a non-zero digit and  $t \geq s + 1$  is minimal with respect to  $d_{t-1} \neq d_t$  ( $t$  is defined for the reduced window). State  $b_3$  occurs iff  $d_{i-1} \neq d_i$  holds and property (7.1) is not fulfilled.

We determine the conditional probabilities  $\Pr(X_{n+1} = b_i | X_n = b_j)$ . First, we consider the case  $X_{n+1} = b_1$  for the different  $X_n$ :

- If the current state is  $X_n = b_1$ :  
That means  $d_{i-1} = d_i$  and the next input bit is  $d_{i-1}$ . Hence,

$$\begin{aligned} \Pr(X_{n+1} = b_1 | X_n = b_1) &= \Pr(d_{i-2} = d_{i-1} | d_{i-1} = d_i) \\ &= \Pr(d_{i-2} = d_{i-1}) = \frac{1}{2}. \end{aligned} \quad (7.2)$$

- If the current state is  $X_n = b_2$ :  
That means  $d_{i-1} \neq d_i$ , property (7.1) is fulfilled, and the next input bit is  $d_{i-w_0-1} = d_{s-1}$ . Obviously, the logical value of  $d_{s-2} = d_{s-1}$  is independent from  $d_{i-1} \neq d_i$  and from property (7.1). Hence,

$$\Pr(X_{n+1} = b_1 | X_n = b_2) = \Pr(d_{s-2} = d_{s-1}) = \frac{1}{2}. \quad (7.3)$$

- If the current state is  $X_n = b_3$ :  
That means  $d_{i-1} \neq d_i$ , property (7.1) is not fulfilled, and the next input bit is  $d_{i-w_0} = d_s$ . As property (7.1) is not fulfilled we must have  $d_{s-1} \neq d_s$ . Hence,

$$\Pr(X_{n+1} = b_1 | X_n = b_3) = \Pr(d_{s-1} = d_s | d_{s-1} \neq d_s) = 0. \quad (7.4)$$

Next, we determine the remaining probabilities  $\Pr(X_{n+1} = b_2 | X_n)$  and  $\Pr(X_{n+1} = b_3 | X_n)$ . If the new state is not  $b_1$ , then  $w_0 + 2$  input bits are processed and the first two of these differ. Therefore there are  $2^{w_0+1}$  possibilities for these input bits. It is a consequence of the MOF properties that for each positive odd integer  $1 \leq x \leq 2^{w_0} - 1$  there are exactly four

choices of  $d_i \neq d_{i-1}, d_{i-2}, \dots, d_s, d_{s-1} \in \{0, 1\}$  such that  $|[d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{t-1} - d_t]_2| = x$  holds, where  $t \geq s$  is minimal with respect to  $d_{t-1} \neq d_t$  (e. g., for  $x = 3$  the four choices are 0110, 1001, 0101 and 1010 leading to the MOF sequences  $10\bar{1}, \bar{1}01, 1\bar{1}\bar{1}$  and  $\bar{1}\bar{1}\bar{1}$ ). Hence, there are  $2^{w_0-1}$  possibilities for the expression  $|[d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{t-1} - d_t]_2|$ , each of which occurs with the same probability. Consequently, we conclude that if the next state is not  $b_1$ , then we have  $X_{n+1} = b_2$  in  $\frac{2^{w_0-2}(1+w_1)}{2^{w_0-1}} = \frac{1+w_1}{2}$  of the cases (namely if  $|[d_{i-1} - d_i, d_{i-2} - d_{i-1}, \dots, d_{t-1} - d_t]_2| \leq 2^{w_0-1}(1+w_1) - 1$  is fulfilled), and we have  $X_{n+1} = b_3$  in the remaining cases, that is  $\frac{1-w_1}{2}$ . Thus, we have proved

$$\Pr(X_{n+1} = b_2 | X_n) = (1 - \Pr(X_{n+1} = b_1 | X_n)) \left( \frac{1+w_1}{2} \right) \quad \text{and} \quad (7.5)$$

$$\Pr(X_{n+1} = b_3 | X_n) = (1 - \Pr(X_{n+1} = b_1 | X_n)) \left( \frac{1-w_1}{2} \right) \quad (7.6)$$

From (7.2), (7.3), (7.5), (7.5), and (7.6) we conclude that the random process is a Markov chain with transition matrix:

	$b_1$	$b_2$	$b_3$
$b_1$	$\frac{1}{2}$	$\frac{1+w_1}{4}$	$\frac{1-w_1}{4}$
$b_2$	$\frac{1}{2}$	$\frac{1+w_1}{4}$	$\frac{1-w_1}{4}$
$b_3$	0	$\frac{1+w_1}{2}$	$\frac{1-w_1}{2}$

As  $w_1 < 1$  holds, the Markov chain is irreducible and aperiodic, and its stationary distribution can be easily determined as

$$\pi = \left( \pi(b_1) = \frac{1+w_1}{3+w_1}, \pi(b_2) = \frac{1+w_1}{3+w_1}, \pi(b_3) = \frac{1-w_1}{3+w_1} \right).$$

The non-zero density of this form is hence

$$\frac{\pi(b_2) + \pi(b_3)}{\pi(b_1)l(b_1) + \pi(b_2)l(b_2) + \pi(b_3)l(b_3)},$$

where  $l(x)$  stands for the length of the block  $x$ . That is:  $\frac{1}{w_0+w_1+1} = \frac{1}{w+1}$ . □

Note that a similar proof can be constructed to show that the non-zero density of *Frac- $w$ NAF* is exactly the same as for *Frac- $w$ MOF* [SSST06]. Moreover, the three states of the Markov process are different, but their mutual dependencies are identical. This supports our claim that *wMOF* and *wNAF* are dual to each other.

**REMARK 7.1** Although Theorem 7.1 as well as Theorem 6.4 is only of asymptotic nature, experimental results published in Okeya *et al.* [OSSST04b] indicate that for all values of cryptographic relevance the provided asymptotic terms reflect the measured real-life values quite accurate<sup>3</sup>.

<sup>3</sup>The data given in [OSSST04b] refers to *wMOF*, but from the generation of *Frac- $w$ MOF* it is clear that the same conclusion can be drawn for the fractional variant.

## 7.3 Comparison

The following theorem is an immediate consequence of the comparison of the standard  $w$ MOF methods with its extended fractional version:

**THEOREM 7.2** *The fractional  $w$ MOF method coincide with its original versions for integer widths (i. e.,  $w_1 = 0$ ).*

Let us now compare the fractional methods with the sliding window method applied to the NAF of the scalar  $d$  (NAF+SW). The following theorem is proved in Schmidt-Samoa *et al.* [SSST06]:

**THEOREM 7.3** *Regarding the number of precomputed elements and the non-zero density, the sliding window method with width  $v$  applied to NAF is equivalent to the fractional  $w$ NAF resp.  $w$ MOF with width  $w = v + \left(\frac{2^{v-2} + (-1)^{v+1}}{3} \frac{1}{2^{v-2}}\right)$ .*

We hence showed that the fractional window schemes are much more flexible than the other methods (standard  $w$ NAF resp.  $w$ MOF, NAF+SW), in the sense that they allow any sizes of precomputed tables. And we proved that they are as efficient as the preceding methods: in the cases where they use tables of the same sizes as standard  $w$ NAF, standard  $w$ MOF, or NAF+SW, they achieve the same non-zero density. This result is illustrated by Fig. 7.2.

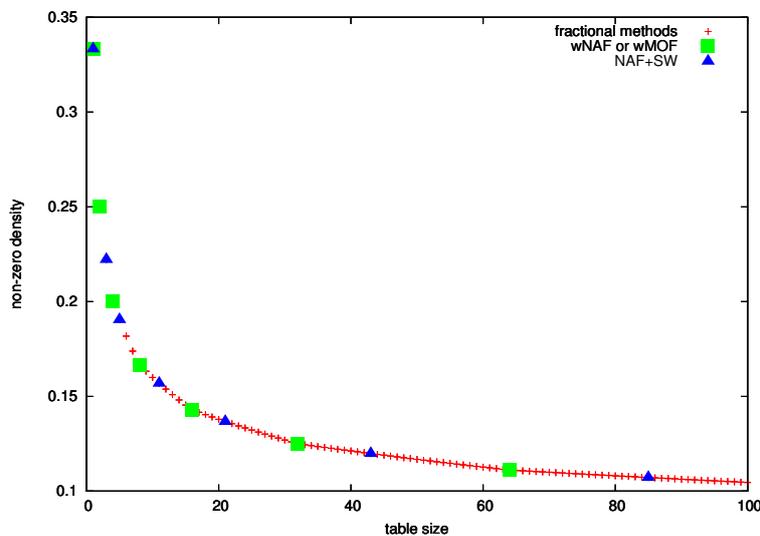


Figure 7.2: Non-zero density and allowed sizes for the precomputed table

Note that in contrast to Fig. 7.1, the gaps between the table sizes are filled now thanks to the fractional methods.

## 7.4 Conclusion

In this chapter, we proposed the fractional  $w$ MOF (Frac- $w$ MOF) method for computing scalar multiplications in elliptic curve cryptosystems (ECC). Frac- $w$ MOF is a left-to-right

analogue to the fractional  $w$ NAF (Frac- $w$ NAF) method. This duality is also highlighted by the observation that both recoding schemes can be modeled as a Markov chain with the same transition matrix. Therefore we used Markov theory to analyze the asymptotic behavior of Frac- $w$ MOF.

We indeed proved that the proposed Frac- $w$ MOF has the same non-zero density as Frac- $w$ NAF using identical table sizes. In addition, our analysis enabled the comparison of the fractional schemes with previously known efficient window methods for EC scalar multiplication. We were able to prove that the main properties of all these methods can be achieved as a special instances of fractional  $w$ MOF resp.  $w$ NAF.

With Frac- $w$ MOF recoding, the whole scalar multiplication can be processed left-to-right, hence requiring less working memory than the Frac- $w$ NAF approach. As a demonstration, we constructed an on-the-fly algorithm for EC scalar multiplication. Since recent results additionally show that Frac- $w$ MOF recoding provides the minimal Hamming weight among all schemes using the same table<sup>4</sup>, we believe that Frac- $w$ MOF is indeed the optimal universally applicable base 2-representation for performing EC scalar multiplications in limited constraint devices.

---

<sup>4</sup>Actually, this result is proved for an even *larger* table including the even elements, too.

# Bibliography

- [ADR02] J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In Lars R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer, 2002.
- [AGK05] S. Abe, R. Gennaro, and K. Kurosawa. Tag-KEM/DEM: A new framework for hybrid encryption. Cryptology ePrint Archive: Report 2005/027, 2005.
- [AM94] L. M. Adleman and K. S. McCurley. Open problems in number theoretic complexity, II. In Leonard M. Adleman and Ming-Deh A. Huang, editors, *ANTS*, volume 877 of *Lecture Notes in Computer Science*, pages 291–322. Springer, 1994.
- [Ava02] R. M. Avanzi. On multi-exponentiation in cryptography, October 2002. For the Advanced Research on Elliptic and Hyperelliptic Curve Cryptography Project; available from <http://www.arehcc.com/download/exporev.pdf>.
- [Ava04] R. M. Avanzi. A note on the signed sliding window integer recoding and a left-to-right analogue. In Helena Handschuh and M. Anwar Hasan, editors, *Selected Areas in Cryptography*, volume 3357 of *Lecture Notes in Computer Science*, pages 130–143. Springer, 2004.
- [Avi61] A. Aviziensis. Signed digit number representations for fast parallel arithmetic. *IRE Trans. Electron. Comput.*, 10:389–400, 1961.
- [BCP03] E. Bresson, D. Catalano, and D. Pointcheval. A simple public-key cryptosystem with a double trapdoor decryption mechanism and its applications. In Lai [Lai03], pages 37–54.
- [BDHG99] D. Boneh, G. Durfee, and N. Howgrave-Graham. Factoring  $N = p^r q$  for large  $r$ . In Wiener [Wie99], pages 326–337.
- [BDPR98] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public-key encryption schemes. In Krawczyk [Kra98], pages 26–45.
- [Bel97] M. Bellare. Practice-oriented provable-security. In Eiji Okamoto, George I. Davida, and Masahiro Mambo, editors, *ISW*, volume 1396 of *Lecture Notes in Computer Science*, pages 221–231. Springer, 1997.
- [BGMW93] E. F. Brickell, D. M. Gordon, K. S. McCurley, and D. B. Wilson. Fast exponentiation with precomputation. In Rueppel [Rue93], pages 200–207.
- [BK90] J. F. Boyar and S. A. Kurtz. A discrete logarithm implementation of perfect zero-knowledge blobs. *Journal of Cryptology*, 2(2):63–76, 1990.
- [Ble98] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In Krawczyk [Kra98], pages 1–12.
- [Boo51] A. D. Booth. A signed binary multiplication technique. *Quarterly Journal of Mechanics and Applied Mathematics*, 4(2):236–240, 1951. Reprinted in E. E. Swartzlander, *Computer Arithmetic*, Vol. 1, IEEE Computer Society Press Tutorial, Los Alamitos, CA, Los Alamitos, CA, 1990.

- [BP97] N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Fumy [Fum97], pages 366 – 377.
- [BP04] M. Bellare and A. Palacio. Towards plaintext-aware public-key encryption without random oracles. In Pil Joong Lee, editor, *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 48–62. Springer, 2004.
- [BR93] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. of the 1st ACM Conference on Computer and Communications Security (CCS)*, pages 62–73. ACM Press, 1993.
- [BR94] M. Bellare and P. Rogaway. Optimal Asymmetric Encryption - how to encrypt with RSA. In Alfredo De Santis, editor, *EUROCRYPT*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer-Verlag, 1994.
- [BR96] M. Bellare and P. Rogaway. The exact security of digital signatures - how to sign with RSA and Rabin. In Ueli M. Maurer, editor, *EUROCRYPT*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer, 1996.
- [Bri93] Ernest F. Brickell, editor. *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*. Springer, 1993.
- [BSS99] I. F. Blake, G. Seroussi, and N. P. Smart. *Elliptic Curves in Cryptography*. Cambridge University Press, 1999. ISBN 0-521-65374-6.
- [BV98] D. Boneh and R. Venkatesan. Breaking RSA may not be equivalent to factoring. In Nyberg [Nyb98], pages 59–71.
- [CF85] J. D. Cohen and M. J. Fischer. A robust and verifiable cryptographically secure election scheme. In *26th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 372–382. IEEE Computer Society Press, 1985.
- [CGH98] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited (preliminary version). In *Proc. of the 30th Annual ACM Symposium on Theory of Computing (STOC '98)*, pages 209–218, New York, NY, USA, 1998. ACM Press.
- [CGH04] R. Canetti, O. Goldreich, and S. Halevi. On the random-oracle methodology as applied to length-restricted signature schemes. In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 40–57. Springer, 2004.
- [CGHGN01] D. Catalano, R. Gennaro, N. Howgrave-Graham, and P. Nguyen. Paillier’s cryptosystem revisited. In *Proc. of the 8th ACM Conference on Computer and Communications Security (CCS)*, pages 206–214. ACM Press, 2001.
- [CMO98] H. Cohen, A. Miyaji, and T. Ono. Efficient elliptic curve exponentiation using mixed coordinates. In Kazuo Ohta and Dingyi Pei, editors, *ASIACRYPT*, volume 1514 of *Lecture Notes in Computer Science*, pages 51–65. Springer, 1998.
- [CNS02] D. Catalano, P. Nguyen, and J. Stern. The hardness of Hensel lifting: The case of RSA and discrete logarithm. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 299–310, Berlin, 2002. Springer-Verlag.
- [CS04] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.*, 33(1):167–226, 2004.
- [DDN91] D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography (extended abstract). In *Proc. of the 23rd Annual ACM Symposium on Theory of Computing (STOC '91)*, pages 542–552, New York, NY, USA, 1991. ACM Press.

- [Den02] A. W. Dent. An implementation attack against the EPOC-2 public-key cryptosystem. *Electronics Letters*, 38(9):412–413, 2002.
- [Des02] Yvo Desmedt, editor. *Public Key Cryptography - PKC 2003, 6th International Workshop on Theory and Practice in Public Key Cryptography, Miami, FL, USA, January 6-8, 2003, Proceedings*, volume 2567 of *Lecture Notes in Computer Science*. Springer, 2002.
- [DH76] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inf. Theory*, IT-22:644–654, 1976.
- [EcKK94] Ö. Egecioglu and Ç. K. Koç. Exponentiation using canonical recoding. *Theor. Comput. Sci.*, 129(2):407–417, 1994.
- [EGM96] S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, 1996.
- [FF02] M. Fischlin and R. Fischlin. The representation problem based on factoring. In Bart Preneel, editor, *CT-RSA*, volume 2271 of *Lecture Notes in Computer Science*, pages 96–113. Springer, 2002.
- [FKM<sup>+</sup>00] E. Fujisaki, T. Kobayashi, H. Morita, H. Oguro, T. Okamoto, S. Okazaki, D. Pointcheval, and S. Uchiyama. EPOC: Efficient probabilistic public-key encryption, August 2000. Submitted to ISO and NESSIE. Available from <http://www.di.ens.fr/~pointche/proposals/ISO/> and <http://www.di.ens.fr/~pointche/proposals/NESSIE/>.
- [FO99a] E. Fujisaki and T. Okamoto. How to enhance the security of public-key encryption at minimum cost. In Imai and Zheng [IZ99], pages 53–68.
- [FO99b] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Wiener [Wie99], pages 537–554.
- [FOM91] A. Fujioka, T. Okamoto, and S. Miyaguchi. ESIGN: An efficient digital signature implementation for smart cards. In Donald W. Davies, editor, *EUROCRYPT*, volume 547 of *Lecture Notes in Computer Science*, pages 446–457, Berlin, 1991. Springer-Verlag.
- [FOPS01] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is secure under the RSA assumption. In Kilian [Kil01], pages 260–274.
- [Fra04] Matthew K. Franklin, editor. *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*. Springer, 2004.
- [Fuj01] E. Fujisaki. Chosen-chipertext security of EPOC-2. Technical report, NTT Corporation, 2001.
- [Fum97] Walter Fumy, editor. *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, volume 1233 of *Lecture Notes in Computer Science*. Springer, 1997.
- [Gam84] T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and David Chaum, editors, *CRYPTO*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer, 1984.
- [GB01] S. Goldwasser and M. Bellare. Lecture notes on cryptography, August 2001. Available from <http://www-cse.ucsd.edu/users/mihir/papers/gb.html>.

- [Gen04] R. Gennaro. Multi-trapdoor commitments and their applications to proofs of knowledge secure under concurrent man-in-the-middle attacks. In Franklin [Fra04], pages 220–236.
- [GLV01] R. P. Gallant, R. J. Lambert, and S. A. Vanstone. Faster point multiplication on elliptic curves with efficient endomorphisms. In Kilian [Kil01], pages 190–200.
- [GM82] S. Goldwasser and S. Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *Proc. of the 14th Annual ACM Symposium on Theory of Computing (STOC '82)*, pages 365–377, New York, NY, USA, 1982. ACM Press.
- [GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
- [GM97] M. Girault and J.-F. Misarsky. Selective forgery of RSA signatures using redundancy. In Fumy [Fum97], pages 495–507.
- [GMMV03] D. Galindo, S. M. Molleví, P. Morillo, and J. L. Villar. A practical public key cryptosystem from Paillier and Rabin schemes. In Desmedt [Des02], pages 279–291.
- [GMR84] S. Goldwasser, S. Micali, and R. L. Rivest. A “paradoxical” solution to the signature problem (extended abstract). In *25th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 441–448. IEEE Computer Society Press, 1984.
- [GMR88] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
- [Gol03] O. Goldreich. Foundations of cryptology, volume ii, 2003. Available from <http://www.wisdom.weizmann.ac.il/~oded/PSBookFrag/enc.ps>.
- [Gor98] D. M. Gordon. A survey of fast exponentiation methods. *Journal of Algorithms*, 27:129–146, 1998.
- [Häg02] O. Häggström. *Finite Markov Chains and Algorithmic Applications*. Cambridge University Press, 2002.
- [HGPT05] C. Heuberger, P. Grabner, H. Prodinger, and J. Thuswaldner. Analysis of linear combination algorithms in cryptography. *ACM Transactions on Algorithms*, 1:123–142, 2005.
- [HKPR05] C. Heuberger, R. Katti, H. Prodinger, and X. Ruan. The alternating greedy expansion and applications to left-to-right algorithms in cryptography. *Theor. Comput. Sci. A*, 341:55–72, 2005.
- [HOQ97] Yongfei Han, Tatsuaki Okamoto, and Sihang Qing, editors. *Information and Communication Security, First International Conference, ICICS'97, Beijing, China, November 11-14, 1997, Proceedings*, volume 1334 of *Lecture Notes in Computer Science*. Springer, 1997.
- [HW79] G. Hardy and E. Wright. *An Introduction to The Theory Of Numbers*. Oxford Press, fifth edition edition, 1979.
- [INS00] INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS (IEEE). Standard specifications for public-key cryptography – ieee p1363-2000, 2000. <http://grouper.ieee.org/groups/1363/>.
- [IZ99] Hideki Imai and Yuliang Zheng, editors. *Public Key Cryptography, Second International Workshop on Practice and Theory in Public Key Cryptography, PKC '99, Kamakura, Japan, March 1-3, 1999, Proceedings*, volume 1560 of *Lecture Notes in Computer Science*. Springer, 1999.
- [JC86] W. De Jonge and D. Chaum. Attacks on some RSA signatures. In Williams [Wil86], pages 18–27.

- [JM89] J. Jedwab and C. J. Mitchell. Minimum weight modified signed-digit representations and fast exponentiation. *Electronics Letters*, 25:1171–1172, 1989.
- [JQY01] M. Joye, J.-J. Quisquater, and M. Yung. On the power of misbehaving adversaries and security analysis of the original EPOC. In Naccache [Nac01], pages 208–222.
- [JY00] M. Joye and S.-M. Yen. Optimal left-to-right binary signed-digit recoding. *IEEE Trans. Computers*, 49(7):740–748, 2000.
- [Kil01] Joe Kilian, editor. *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*. Springer, 2001.
- [KIT88] K. Kurosawa, T. Ito, and M. Takeuchi. Public key cryptosystem using a reciprocal number with the same intractability as factoring a large number. *CRYPTOLOGIA*, 12(4):225–233, 1988.
- [Knu81] D. E. Knuth. *The art of computer programming, Volume 2: Seminumerical Algorithms*. Addison-Wesley, Reading, Mass., 2 edition, 1981.
- [Kob87] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.
- [KR00] H. Krawczyk and T. Rabin. Chameleon signatures. In *Proc. of the Symposium on Network and Distributed Systems Security (NDSS)*. The Internet Society, 2000.
- [Kra98] Hugo Krawczyk, editor. *Advances in Cryptology - CRYPTO '98, 18th Annual International Cryptology Conference, Santa Barbara, California, USA, August 23-27, 1998, Proceedings*, volume 1462 of *Lecture Notes in Computer Science*. Springer, 1998.
- [KSST03] K. Kurosawa, K. Schmidt-Samoa, and T. Takagi. A complete and explicit security reduction algorithm for rsa-based cryptosystems. In Laih [Lai03], pages 474–491.
- [KT92] K. Koyama and Y. Tsuruoka. Speeding up elliptic cryptosystems by using a signed binary window method. In Brickell [Bri93], pages 345–357.
- [KYLH94] K.-Y-Lam and L. C. K. Hui. Efficiency of SS(1) square-and-multiple exponentiation algorithms. *Electronic Letters*, 30:2115–2116, 1994.
- [Lai03] Chi-Sung Laih, editor. *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings*, volume 2894 of *Lecture Notes in Computer Science*. Springer, 2003.
- [Len87] H. W. Lenstra, Jr.. Factoring integers with elliptic curves. *Annals of Mathematics*, 126:649–673, 1987.
- [LL93] A. K. Lenstra and H. W. Lenstra, Jr., editors. *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*. Springer-Verlag, 1993.
- [LL94] C. H. Lim and P. J. Lee. More flexible exponentiation with precomputation. In Yvo Desmedt, editor, *CRYPTO*, volume 839 of *Lecture Notes in Computer Science*, pages 95–107. Springer, 1994.
- [LL03] Pil Joong Lee and Chae Hoon Lim, editors. *Information Security and Cryptology - ICISC 2002, 5th International Conference Seoul, Korea, November 28-29, 2002, Revised Papers*, volume 2587 of *Lecture Notes in Computer Science*. Springer, 2003.
- [MG02] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, 2002.
- [Mil85] V. S. Miller. Use of elliptic curves in cryptography. In Williams [Wil86], pages 417–426.

- [MK04] A. J. Menezes and N. Kobitz. Another look at “provable security”. Cryptology ePrint Archive, Report 2004/152, 2004. <http://eprint.iacr.org/>.
- [MO90] F. Morain and J. Olivos. Speeding up the computations on an elliptic curve using addition-subtraction chains. *RAIRO Theoretical Informatics and Applications*, 24:531–543, 1990.
- [MOC97] A. Miyaji, T. Ono, and H. Cohen. Efficient elliptic curve exponentiation. In Han et al. [HOQ97], pages 282–290.
- [Möl01] B. Möller. Securing elliptic curve point multiplication against side-channel attacks. In George I. Davida and Yair Frankel, editors, *ISC*, volume 2200 of *Lecture Notes in Computer Science*, pages 324–334. Springer, 2001.
- [Möl02] B. Möller. Improved techniques for fast exponentiation. In Lee and Lim [LL03], pages 298–312.
- [Möl04] B. Möller. Fractional windows revisited: Improved signed-digit representations for efficient exponentiation. In Choonsik Park and Seongtaek Chee, editors, *ICISC*, volume 3506 of *Lecture Notes in Computer Science*, pages 137–153. Springer, 2004.
- [MS05] J. A. Muir and D. R. Stinson. New minimal weight representations for left-to-right window methods. In Alfred J. Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 366–383. Springer, 2005.
- [MS06] J. A. Muir and D. R. Stinson. Minimality and other properties of the width- $w$  nonadjacent form. *Mathematics of Computation*, 75:369–384, 2006.
- [Mui04] J. A. Muir. *Efficient Integer Representations for Cryptographic Operations*. PhD thesis, University of Waterloo, December 2004.
- [MvOV96] A. J. Menezes, P. C. van Oorschot, and S. A. Vanston. *Handbook of Applied Cryptography*. CRC Press, 1996. Available from <http://www.cacr.math.uwaterloo.ca/hac/>.
- [Nac01] David Naccache, editor. *Topics in Cryptology - CT-RSA 2001, The Cryptographer’s Track at RSA Conference 2001, San Francisco, CA, USA, April 8-12, 2001, Proceedings*, volume 2020 of *Lecture Notes in Computer Science*. Springer, 2001.
- [New04] Ist-1999-12324, April 2004. <https://www.cosic.esat.kuleuven.ac.be/nessie/>.
- [NS98] D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *Proc. of the 5th ACM Conference on Computer and Communications Security (CCS)*, pages 59–66. ACM Press, 1998.
- [Nyb98] Kaisa Nyberg, editor. *Advances in Cryptology - EUROCRYPT ’98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*. Springer, 1998.
- [NZ76] I. Niven and H. S. Zuckerman. *Einführung in die Zahlentheorie*. B.I.-Wissenschaftsverlag, 1976.
- [OP00] T. Okamoto and D. Pointcheval. EPOC-3 - efficient probabilistic public-key encryption, 2000. Submitted to IEEE P1363.
- [OP01] T. Okamoto and D. Pointcheval. REACT: Rapid enhanced-security asymmetric cryptosystem transform. In Naccache [Nac01], pages 159–175.
- [OSSST04a] K. Okeya, K. Schmidt-Samoa, C. Spahn, and T. Takagi. Signed binary representations revisited. In Franklin [Fra04], pages 123–139.
- [OSSST04b] K. Okeya, K. Schmidt-Samoa, C. Spahn, and T. Takagi. Signed binary representations revisited. Cryptology ePrint Archive: Report 2004/195, 2004. <http://eprint.iacr.org/>.

- [OU98] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In Nyberg [Nyb98], pages 308–318.
- [Pai99a] P. Paillier. Public key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT*, volume 1592 of *Lecture Notes in Computer Science*, pages 223 – 238. Springer-Verlag, 1999.
- [Pai99b] P. Paillier. A trapdoor permutation equivalent to factoring. In Imai and Zheng [IZ99], pages 219 – 222.
- [PG97] J. Patarin and L. Goubin. Trapdoor one-way permutations and multivariate polynomials. In Han et al. [HOQ97], pages 356–368.
- [PO96] R. Peralta and E. Okamoto. Faster factoring of integers of a special form. *TIEICE: IEICE Transactions on Communications/Electronics/Information and Systems*, E79-A(4):489–493, 1996.
- [Poi04] D. Pointcheval. Provable security for public key schemes. Advanced Course on Contemporary Cryptology, CRM, Barcelona, Spain, February 2004. Available from <http://www.di.ens.fr/~pointche/pub.php?reference=Po04>.
- [PP97] T. P. Pedersen and B. Pfitzmann. Fail-stop signatures. *SIAM J. Comput.*, 26(2):291–330, 1997.
- [Pro00] H. Prodinger. On binary representations of integers with digits  $-1, 0, 1$ . *Integers. Electronic Journal of Combinatorial Number Theory*, 2000.
- [PW90] B. Pfitzmann and M. Waidner. Formal aspects of fail-stop signatures. Technical report, Universität Karlsruhe, 1990.
- [QK02] J-J Quisquater and F. Koeune. Side channel attacks – state of the art, October 2002. Available from [http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1047\\_Side\\_Channel\\_report.pdf](http://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1047_Side_Channel_report.pdf).
- [Rab79] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report 212, MIT Laboratory of Computer Science, Cambridge, 1979.
- [Rei60] G. W. Reitwiesner. Binary arithmetic. In *Advances in Computers*, volume 1, pages 231–308. Academic Press, 1960.
- [RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- [Rue93] Rainer A. Rueppel, editor. *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings*, volume 658 of *Lecture Notes in Computer Science*. Springer, 1993.
- [SCA] [http://www.crypto.ruhr-uni-bochum.de/en\\_sclounge.html](http://www.crypto.ruhr-uni-bochum.de/en_sclounge.html).
- [Sha48] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, July , October 1948.
- [Sha49] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4), October 1949.
- [Sho99] Victor Shoup. On the security of a practical identification scheme. *J. Cryptology*, 12(4):247–260, 1999.
- [Sho02] V. Shoup. OAEP reconsidered. *J. Cryptology*, 15(4):223–249, 2002.
- [Sho04a] V. Shoup. Draft iso 18033-2, December 2004. <http://shoup.net/iso/>.

- [Sho04b] V. Shoup. Sequences of games: a tool for taming complexity in security proofs. Available from <http://shoup.net/papers/>, November 2004.
- [Sol00] J. A. Solinas. Efficient arithmetic on Koblitz curves. *Designs, Codes and Cryptography*, 19:195–249, 2000.
- [SS04] K. Schmidt-Samoa. Factorization-based fail-stop signatures revisited. In Javier Lopez, Sihan Qing, and Eiji Okamoto, editors, *ICICS*, volume 3269 of *Lecture Notes in Computer Science*, pages 118–131. Springer, 2004.
- [SS06] K. Schmidt-Samoa. A new Rabin-type trapdoor permutation equivalent to factoring and its applications. *Electr. Notes Theor. Comput. Sci.*, xxx:xx–xx, 2006.
- [SSN03] W. Susilo and R. Safavi-Naini. An efficient fail-stop signature scheme based on factorization. In Lee and Lim [LL03], pages 62 – 74.
- [SSNGS00] W. Susilo, R. Safavi-Naini, M. Gysin, and J. Seberry. A new and efficient fail-stop signature scheme. *The Computer Journal*, 43(5):430–437, 2000.
- [SSNP99] W. Susilo, R. Safavi-Naini, and J. Pieprzyk. RSA-based fail-stop signature schemes. In *ICPP Workshop*, pages 161–166, 1999.
- [SSST06] K. Schmidt-Samoa, O. Semay, and T. Takagi. Analysis of fractional window methods and their application to elliptic curve cryptosystems. *IEEE Trans. Computers*, 2006. to appear.
- [SST05] K. Schmidt-Samoa and T. Takagi. Paillier’s cryptosystem modulo  $p^2q$  and its applications to trapdoor commitment schemes. In Ed Dawson and Serge Vaudenay, editors, *Mycrypt*, volume 3715 of *Lecture Notes in Computer Science*, pages 296–313. Springer, 2005.
- [ST01] A. Shamir and Y. Tauman. Improved online/offline signature schemes. In Kilian [Kil01], pages 355–367.
- [ST02] K. Sakurai and T. Takagi. New semantically secure public-key cryptosystems from the RSA primitive. In David Naccache and Pascal Paillier, editors, *Public Key Cryptography*, volume 2274 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2002.
- [ST03] K. Sakurai and T. Takagi. A reject timing attack on an IND-CCA2 public-key cryptosystem. In Lee and Lim [LL03], pages 359–379.
- [Sti90] D. R. Stinson. Some observations on parallel algorithms for fast exponentiation in  $GF(2^n)$ . *SIAM J. Comput.*, 19(4):711–717, 1990.
- [Sti04] D. R. Stinson. A polemic on notions of cryptographic security, July 28 2004. Available from <http://www.cacr.math.uwaterloo.ca/~dstinson/papers/polemic.ps>.
- [Tak98] T. Takagi. Fast RSA-type cryptosystem modulo  $p^kq$ . In Krawczyk [Kra98], pages 318–326.
- [Tak04] T. Takagi. A fast RSA-type public-key primitive modulo  $p^kq$  using Hensel lifting. *IEICE Transactions*, Vol.E87-A(1):94–101, 2004.
- [vHP93] E. van Heyst and T. P. Pedersen. How to make efficient fail-stop signatures. In Rueppel [Rue93], pages 366 – 377.
- [vHPP92] E. van Heijst, T. P. Pedersen, and B. Pfitzmann. New constructions of fail-stop signatures and lower bounds (extended abstract). In Brickell [Bri93], pages 15–30.
- [vzG91] J. v. z. Gathen. Efficient and optimal exponentiation in finite fields. *Computational Complexity*, 1:360–394, 1991.

- [Wie99] Michael J. Wiener, editor. *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, volume 1666 of *Lecture Notes in Computer Science*. Springer, 1999.
- [Wil86] Hugh C. Williams, editor. *Advances in Cryptology - CRYPTO '85, Santa Barbara, California, USA, August 18-22, 1985, Proceedings*, volume 218 of *Lecture Notes in Computer Science*. Springer, 1986.
- [WMPW98] E. De Win, S. Mister, B. Preneel, and M. J. Wiener. On the performance of signature schemes based on elliptic curves. In Joe Buhler, editor, *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 252–266. Springer, 1998.
- [WSI03] Y. Watanabe, J. Shikata, and H. Imai. Equivalence between semantic security and indistinguishability against chosen ciphertext attacks. In Desmedt [Des02], pages 71–84.
- [Yao76] A. C.-C. Yao. On the evaluation of powers. *SIAM J. Comput.*, 5:100–103, 1976.

# Index

- adaptive chosen-message attack, 18
- altering
  - strong, 74
  - weak, 73
- binary method, 84
- black box reduction, 13
- Blum-Williams function, 40
- Booth algorithm, 96
- bundling degree, 66
- bundling homomorphism, 63, **65**
- carry-free  $w$ NAF, 97
- chameleon signature, 75
- Charmichael's function, 11
- chosen-ciphertext attack, 13
- chosen-plaintext attack, 13
- collision resistance, 66, 74
- computational security, 7
- concrete security, 10
- data encapsulation mechanism, 53
- Decisional Composite Residuosity Assumption, 16, **45**
- DEM, 53
- digital signature scheme, 17
  - chameleon, 75
  - fail-stop, 62
  - on-line/offline, 74
  - RSA, 17
- EPOC, 9, **55**, 61
- Euclidean reduction, 33
- Euler's totient function, 11
- existential forgery, 18
- fail-stop signature scheme, 62
- fault attacks, 10
- forgery
  - existential, 18
  - selective, 18
- Frac- $w$ MOF, 106, **109**
- Frac- $w$ NAF, 106, 107, **108**
- fractional  $w$ MOF, 106
- fractional  $w$ NAF, 106, 107
- full domain hash, 19
- Gaussian reduction algorithm, 25
- generic chosen message attack, 18
- Hamming weight, 87, 105
- hash function, 19
- hash functions, 8
- Hensel-lifting problem, 26
- homomorphic encryption, 13, 47, **48**
- hybrid encryption, 53
- IND-CCA, 15
- IND-CCA1, 16
- IND-CCA2, 16
- IND-CPA, 15, 55
- indistinguishability, 15
- information-theoretic security, 7
- KEM, 53
  - Tag-, 53
- key-encapsulation mechanism, 53
- key-only attack, 17
- known-message attack, 18
- $L$ -function, 11
- lunchtime attack, 16, 24
- Markov theory, 106
- message attack, 18
- MOF, **94**, 95
- $n$ -th residue, 43
- NAF, 87, **87**
  - windowed, 88, 113
- negligible function, 13

- no message attack, 17
- non-malleability, 16
- OAEP conversion, 21, 23
- Okamoto-Uchiyama encryption scheme, **48**, 61
- on-line/off-line signature, 74
- one-time security, 55, 65
- one-way function, 39
- one-wayness, 14
  - partial-domain, 24
- OW-CCA, 14
- OW-CPA, 14
  
- Paillier's encryption scheme, **12**, 16
- partial-domain one-wayness, 24
- perfect secrecy, 7, 15
- plaintext awareness, 23
  - weakly, 23
- plaintext-checking attack, 55
- $p^2q$  Factorization Assumption, 43
- probabilistic encryption, 12
- public key encryption scheme, 11
  - EPOC-3, 55
  - Okamoto-Uchiyama, 48
  - Paillier, 12
  - RSA, 12
  - RSA-OAEP, 23
  - RSA-Paillier, 26
  
- quadratic residue, 40
  
- Rabin-type modular squaring, 40
- Rabin-Williams function, 40
- random oracle model, **8**, 21, 53, 57
- REACT, **54**, 60
- reductionist security, 8
- relaxed uniformity, 78
- RSA assumption, 14
- RSA encryption scheme, **12**, 40
- RSA signature scheme, **17**, 19
- RSA-OAEP, 9, 35
- RSA-Paillier encryption scheme, 26, 36
- RSA-PSS, 17
  
- scalar multiplication, 83
- scalar recoding, 84
- selective forgery, 18
- semantic security, 15
  
- Shannon, 7, 15
- side channel attacks, 10, 86
- signcryption, 75
- signed binary representation, 86
- signed representation, 85
- sliding window methods, 86
- standard model, 9, 21
- stationary distribution, 107
  
- $\mathcal{T}$ -representation, 84
- Tag-KEM, **55**
- Tag-KEM/DEM framework, 53
- tight security reduction, 9
- total break, 18
- transition matrix, 107
- trapdoor commitment scheme, 73
- trapdoor hash family, 73
- trapdoor one-way function, 39, **41**
- trapdoor one-way permutation, 23, 39, **41**
  
- UF-CMA, **18**
- uniformity, 74
  - relaxed, 78
  
- Vernam's one-time pad, 7, 53
  
- window methods, 86
  - fractional, 105
  - sliding, 86
- $w$ MOF, 99, **99**, 100, 106
  - fractional, 106, 107, 109
- $w$ NAF, 87, **89**, 90, 96, 97, 106
  - carry-free, 97
  - fractional, 106
  
- $x$ -minimal condition, 27
- $x$ -minimal solution, 27
  
- zero-minimum condition, 29