Lattice-based Signature Schemes with Additional Features

Vom Fachbereich Informatik der
Technischen Universität Darmstadt genehmigte

Dissertation

zur Erlangung des Grades
Doktor rerum naturalium (Dr. rer. nat.)

von

Dipl.-Inform. Markus Rückert

geboren in Darmstadt.

Referenten:	Prof. Dr. Johannes Buchmann
	Prof. Dr. Daniele Micciancio
Tag der Einreichung:	03. November 2010
Tag der mündlichen Prüfung:	20. Dezember 2010
Hochschulkennziffer:	D 17

Darmstadt 2011

To my family.

Wissenschaftlicher Werdegang

Oktober 2008 – heute

Wissenschaftlicher Mitarbeiter am Lehrstuhl von Prof. Dr. Johannes Buchmann, Fachbereich Informatik, Fachgebiet Theoretische Informatik — Kryptographie und Computeralgebra, Technische Universität Darmstadt.

Juli 2008 – heute

Wissenschaftlicher Mitarbeiter im Projekt „Kryptographische Primitive“ des Arbeitsbereichs „Sichere Daten“ im Center for Advanced Security Research Darmstadt (CASED).

November 2007 – September 2008

Wissenschaftlicher Mitarbeiter am Darmstädter Zentrum für IT-Sicherheit (DZI).

Oktober 2002 – April 2007

Studium der Informatik mit Schwerpunkt Theoretische Informatik und Nebenfach Betriebswirtschaftslehre an der Technischen Universität Darmstadt.

List of Publications

[PUB1] Johannes Buchmann, Richard Lindner, and Markus Rückert. Explicit hard instances of the shortest vector problem. In Johannes Buchmann and Jintai Ding, editors, *PQCrypto*, volume 5299 of *Lecture Notes in Computer Science*, pages 79–94. Springer, 2008.

[PUB2] Johannes Buchmann, Richard Lindner, Markus Rückert, and Michael Schneider. Explicit hard instances of the shortest vector problem (extended version). Cryptology ePrint Archive, Report 2008/333, 2008. `http://eprint.iacr.org/`.

[PUB3] Johannes Buchmann, Richard Lindner, Markus Rückert, and Michael Schneider. Post-quantum cryptography: lattice signatures. *Computing*, 85(1-2):105–125, 2009.

[PUB4] Markus Rückert and Dominique Schröder. Aggregate and verifiably encrypted signatures from multilinear maps without random oracles. In Jong Hyuk Park, Hsiao-Hwa Chen, Mohammed Atiquzzaman, Changhoon Lee, Tai-Hoon Kim, and Sang-Soo Yeo, editors, *ISA*, volume 5576 of *Lecture Notes in Computer Science*, pages 750–759. Springer, 2009.

[PUB5] Markus Rückert and Dominique Schröder. Security of verifiably encrypted signatures and a construction without random oracles. In Hovav Shacham and Brent Waters, editors, *Pairing*, volume 5671 of *Lecture Notes in Computer Science*, pages 17–34. Springer, 2009.

[PUB6] Markus Rückert. Verifiably encrypted signatures from RSA without NIZKs. In Bimal K. Roy and Nicolas Sendrier, editors, *INDOCRYPT*, volume 5922 of *Lecture Notes in Computer Science*, pages 363–377. Springer, 2009.

[PUB7] Markus Rückert and Dominique Schröder. Fair partially blind signatures. In Daniel J. Bernstein and Tanja Lange, editors, *AFRICACRYPT*, volume 6055 of *Lecture Notes in Computer Science*, pages 34–51. Springer, 2010.

[PUB8] Markus Rückert. Strongly unforgeable signatures and hierarchical identity-based signatures from lattices without random oracles. In Nicolas Sendrier, editor, *PQCrypto*, volume 6061 of *Lecture Notes in Computer Science*, pages 182–200. Springer, 2010.

[PUB9] Markus Rückert, Michael Schneider, and Dominique Schröder. Generic constructions for verifiably encrypted signatures without random oracles or nizks. In Jianying Zhou and Moti Yung, editors, *ACNS*, volume 6123 of *Lecture Notes in Computer Science*, pages 69–86, 2010.

[PUB10] Markus Rückert. Adaptively secure identity-based identification from lattices without random oracles. In Juan A. Garay and Roberto De Prisco, editors, *SCN*, volume 6280 of *Lecture Notes in Computer Science*, pages 345–362. Springer, 2010.

[PUB11] Pierre-Louis Cayrel, Richard Lindner, Markus Rückert, and Rosemberg Silva. A lattice-based threshold ring signature scheme. In Michel Abdalla and Paulo S. L. M. Barreto, editors, *LATINCRYPT*, volume 6212 of *Lecture Notes in Computer Science*, pages 255–272. Springer, 2010.

[PUB12] Pierre-Louis Cayrel, Richard Lindner, Markus Rückert, and Rosemberg Silva. Improved zero-knowledge identification with lattices. In Swee-Huay Henc and Kaoru Kurosawa, editors, *ProvSec*, volume 6402 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2010.

[PUB13] Markus Rückert. Lattice-based blind signatures. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 413–430. Springer, 2010.

[PUB14] Markus Rückert and Michael Schneider. Estimating the security of lattice-based cryptosystems. Cryptology ePrint Archive, Report 2010/137, 2010. `http://eprint.iacr.org/`.

[PUB15] Jan Camenisch, Gregory Neven, and Markus Rückert. Lattice-based group signatures from anonymous attribute tokens. Manuscript, 2010.

Acknowledgments

First and foremost, I thank my supervisor Johannes Buchmann for supporting and challenging me throughout the last three years. He kindly let me pursue my interest in lattices, which was first stirred by Alexander May. Furthermore, I am grateful for having Daniele Micciancio as well as Sorin Huss, Marc Fischlin, and Mark Manulis on my PhD committee.

My research topic has put me in a fascinating environment with a very supportive community and I am particularly thankful to Jan Camenisch, Marc Fischlin, Benoît Libert, Vadim Lyubashevsky, Daniele Micciancio, "Gemeindepräsident" Gregory Neven, Chris Peikert, Oded Regev, Nigel Smart, and Bogdan Warinschi for countless insightful and informative discussions — I appreciate your patience with a rookie! Also, I thank ECRYPT II and CASED for their financial support.

I thank my co-authors for their help, encouragement, and cooperation: Johannes Buchmann [PUB1, PUB2, PUB3], Jan Camenisch [PUB15], Pierre-Louis Cayrel [PUB11, PUB12], Richard Lindner [PUB1, PUB2, PUB3, PUB11, PUB12], Gregory Neven [PUB15], Michael Schneider [PUB2, PUB3, PUB9], Dominique Schröder [PUB4, PUB5, PUB7, PUB9], and Rosemberg Silva [PUB11, PUB12].

Special thanks go to my dear CDC, Minicrypt, and CASED colleagues. You have made me enjoy work (and leasure) and many of you have helped me to improve this document. I indebted to Dominique Schröder for being a spark that made me start my own research. I appreciate the many hours we've spent in front of the whiteboard. If it was not for Michael Kreutzer, I would not have had a position in 2007. Without Marita and Cornelia, things would certainly not run as smoothly and efficiently as they (often unnoticed) do.

Moreover, I appreciate the great support of my family and in-laws. Having loyal friends has always been very important to me. Thank you, Bluck, Chris, Domi, Flo, Heike, Markus, Michi, Sandra, Simon, Tobi for all these little, compensatory distractions. Finally, I am indebted to my beautiful wife for her patience, love, and inspiration.

Darmstadt, November 2010 *Markus Rückert*

Zusammenfassung

Nahezu die gesamte kryptographische Landschaft basiert heute auf der Unlösbarkeit zweier Probleme — dem Faktorisierungsproblem und dem Problem diskrete Logarithmen zu berechnen. Diese Duokultur könnte im Falle neuartiger Angriffe rasch zum Kollaps ganzer Teilbereiche der modernen Kryptographie führen. Die konkrete Bedrohung durch Quantencomputer, beispielsweise mit Shor's Faktorisierungsalgorithmus (Shor 1997), ist hierbei nur ein Grund nach Alternativen zu suchen.

Mit Gitterproblemen, wie dem Problem sehr kurze Gittervektoren zu finden, steht bereits eine gut erforschte Alternative zur Verfügung. Diese Probleme zeigen sich resistent gegenüber Quantencomputern und sie widerstehen, im Gegensatz zum Faktorisierungsproblem, subexponentiellen Algorithmen.

In der Gitterkryptographie kann man auf sehr milde Annahmen zurückgreifen und damit außergewöhnlich starke Sicherheitsgarantien erzielen. Ajtais Entdeckung der unterliegenden komplexitätstheoretischen „worst-case to average-case" Reduktion (Ajtai 1996) besagt, dass eine *zufällige* Instanz bestimmter Gitterprobleme mindestens so schwer ist wie die *schwierigste* Instanz eines verwandten Problems.

Mit den Arbeiten (Gentry, Peikert, Vaikuntanathan 2008), (Lyubashevsky, Micciancio 2008), (Lyubashevsky 2009) und (Cash, Hofheinz, Kiltz, Peikert 2009) zu Signaturverfahren stehen solide Grundbausteine zur Verfügung. Möchte man diese in Geschäftsprozessen nutzen, greifen sie jedoch häufig zu kurz. Oft sind hier Zusatzeigenschaften notwendig, um auf Signaturen Berechnungen ausführen zu können.

Die vorliegende Dissertation zeigt die Vielseitigkeit von Gittern in der Kryptographie anhand ausgewählter Anwendungsszenarien auf. Mit den vorgestellten Verfahren unterstützt sie elektronische Wahlverfahren, Vertragsunterzeichnung über das Internet sowie die Signaturkompression. Des Weiteren werden Techniken zur Erfüllung des stärksten Sicherheitsmodells für Signaturverfahren ohne vereinfachende Annahmen, wie Random Oracles, vorgestellt und diskutiert, wie sich identitätsbasierte Primitive verwirklichen lassen. Es ist zu erwarten, dass sich die vorgestellten Techniken verallgemeinern und auf andere Anwendungsbereiche übertragen lassen.

Unabhängig davon wird die praktische Schwierigkeit der relevanten Gitterprobleme untersucht. Dies ermöglicht die Bewertung und Auswahl sicherer Parametersätze für die gesamte moderne Gitterkryptographie.

Abstract

Building cryptographic schemes upon as many fundamentally different hard problems as possible, seems to be the best way to hedge against future threats such as quantum computers. Being mainly based on the hardness of factoring and computing discrete logarithms, the present security landscape is at risk.

In contrast, problems in lattices, such as finding short non-zero vectors, seem to withstand quantum computer attacks and the best known algorithms run in exponential time. In sharp contrast to other fields of cryptography, lattices admit a worst-case to average-case reduction (Ajtai 1996). Instead of assuming that a problem is hard for randomly chosen instances, lattice-based cryptosystems merely require the existence of a single hard instance, i.e., hardness in the worst case. With such an additional "trust anchor", the resulting security guarantees are much more plausible.

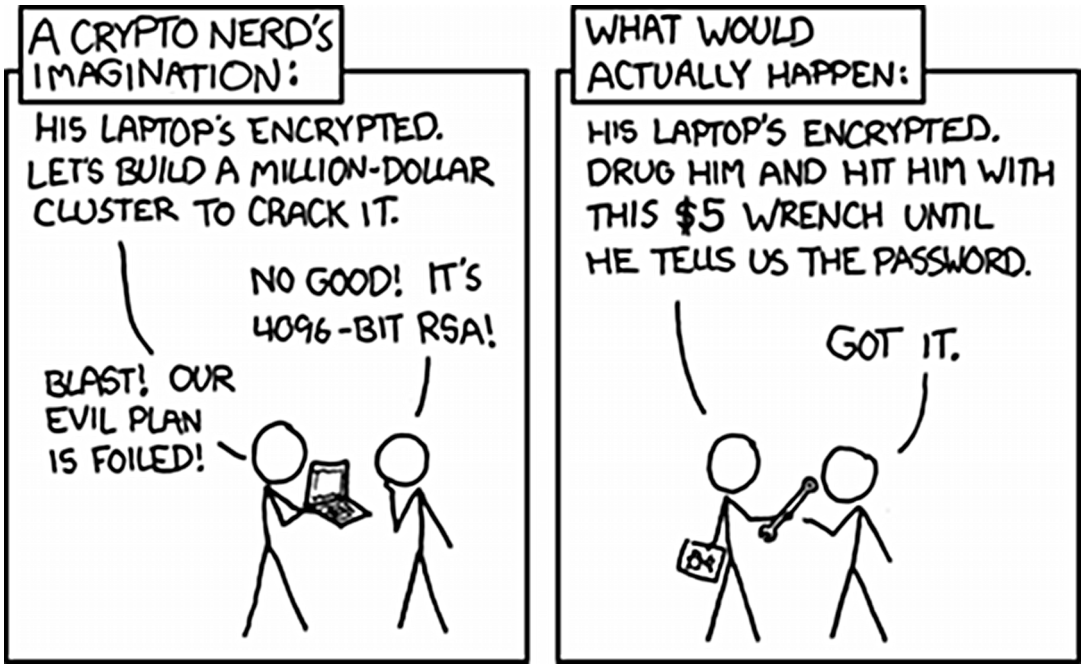
Quite recently, we have seen an increased interest in lattice-based cryptography with many striking results. In this thesis, we are particularly interested in signature schemes, which provide a supporting pillar for today's economy. While we have seen basic signature schemes from lattices, e.g., (Gentry, Peikert, Vaikuntanathan 2008), (Lyubashevsky, Micciancio 2008), (Lyubashevsky 2009), or (Cash, Hofheinz, Kiltz, Peikert 2009), there are hardly any results dealing with the specific needs of applications, where ordinary signatures often fall too short.

In this thesis, we build upon the above results and equip them with additional features, motivated by an exemplary selection of application scenarios. Hence, we demonstrate the great versatility of lattices in cryptography.

In particular, we facilitate privacy-friendly electronic elections, fair online contract signing, signature compression, secure signatures in the strongest sense, as well as identity-based primitives. As far as possible, we avoid simplifying assumptions, such as the random oracle model. We believe that our techniques can be transferred to other application scenarios as well.

Independently of the these results, we discuss the practical hardness of lattice problems and provide a framework for estimating the security of essentially all modern lattice-based cryptography.

Contents



courtesy of xkcd.com

Chapter 1

Introduction

Digital signature schemes are the cornerstone of e-business, e-government, software security, and many more applications. Meeting just the requirement of replacing a hand-written signature to ensure authenticity is often not enough. Specific applications, such as electronic voting, electronic cash, identity management, archiving, or contract signing, call for additional features that, abstractly speaking, allow us to perform interesting computations on signatures. For that reason, we have seen a plethora of special-purpose signature schemes in the past, with [Cha82, CL01, BGLS03, BMW03, BN06, BKM09] only being a brief selection.

Their importance and economic value are expected to grow in the future as more and more everyday tasks and processes are computerized. As for the near future, we are convinced that currently deployed factoring and discrete logarithm based instantiations are efficient and secure — *but for how long?*

Today, when designing cryptographic schemes, one also has to anticipate emerging technologies that may lead to new attacks. Using quantum computers to apply Shor's polynomial-time factoring algorithm [Sho97] is only one such example and it has become a metaphor for unexpected, threatening developments.

In general, cryptographers are encouraged to use the mildest possible assumptions in complexity theoretic security reductions to try and hedge against such unforeseen attacks. One way of doing so, is to provably base cryptography on so-called worst-case problems, rather than on average-case problems. To see the difference, let us consider the discrete logarithm problem (DLP) in a group $\langle g \rangle = G$ of order n and let DS be a signature scheme that is provably as hard to break as solving DLP. More precisely, a forger for DS can also find a, when challenged with $A = g^a \in G$. To trust in the security proof, we need to directly assume the hardness of DLP in this specific, typically randomly chosen, group G. Unfortunately, there is no sound way to choose a *provably hard* group from the family of all groups of order n. One may argue that the well-known random self-reducibility of DLP allows us to choose a random instance that is as hard to solve as any other instance. While such self-reduction is an asset, it is not the solution to our problem because it keeps the group fixed and merely randomizes the challenge A.

Using lattices, we can use the even stronger tool of a worst-case to average-case reduction, which was discovered by Ajtai [Ajt96]. Informally, it states that finding non-zero short vectors in a *random* lattice of dimension $n \log(n)$ is at least as hard as finding short non-zero vectors in *all*, even the hardest possible, lattices of dimension n. If it existed, a DLP analogy would be to prove that a *random* DLP instance A in a *random* group G of order $n \log(n)$ is at least as hard to solve as the worst-case instance B in the worst-case group H of order n. However, it seems that only lattice problems have such a remarkable trait.

Apart from this major theoretical distinction, lattice problems seem to be much harder to solve than, say, the factoring problem. While factoring is possible in sub-exponential time in the bit-length n of the input, solving cryptographically relevant lattice problems in dimension n requires at least single-exponential time in n [MV10a]. This can be of great practical value when selecting secure parameters for the derived cryptosystems because we will inevitably have to increase n over time and a slower increase preserves efficiency.

When using the subclass of *ideal lattices*, we can find very compact representations of a lattice, while having almost the same strong security arguments [Mic07, LM06, PR06, LPR10]. In this setting, lattice-based schemes are asymptotically efficient, often requiring an essentially linear amount of space and time in the main parameter n. For the same security level, the classical alternatives are much worse. In practice, however, the current situation is different and classical number theoretic schemes typically outperform their lattice-based counterparts. On the positive side, using cryptographic lattices mainly involves simple linear algebra operations over rather small fields that allow word-size arithmetic. Hence, there is not need for special co-processors or long integer arithmetic.

1.1 Related Work

To specify the scope of this work, we define a a cryptographic scheme to be *lattice-based* if its security can be entirely based on the worst-case hardness of lattice problems via Ajtai's reduction [Ajt96] or its descendants [GPV08, LM06, PR06, Reg09, LPR10]. In particular, this rules out related ad-hoc constructions, such as NTRU [HPS98].

Driven by the above prospects, lattice-based cryptography has flourished in the recent years. Since Ajtai's work and its early applications [GGH96, AD97, AD07], the field has been diverging in agreement with two of Impagliazzo's famous "worlds" [Imp95] — *Minicrypt* and *Cryptomania*.

Lattice-based Minicrypt, i.e., cryptography that can be built upon one-way functions, uses the following short integer solution (SIS) problem as the main assumption: On input a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, find a non-zero vector $\mathbf{x} \in \mathbb{Z}^m$ such that $\mathbf{A}\mathbf{x} \equiv \mathbf{0} \pmod{q}$ and $\|\mathbf{x}\| \leq \nu$ for some norm bound ν [Ajt96]. In particular, we have seen one-way functions [Mic07], hash functions [LMPR08], identification schemes [Lyu08a, KTX08], and signature schemes [LM08, GPV08, SSTX09, Lyu09, CHKP10, Boy10]. See also [MR08, BLRS09] for an overview. While ordinary signature schemes are in Minicrypt, this classification does not necessarily hold for our

subject. Occasionally, the desired additional features require techniques, such as trapdoor functions, from Cryptomania.

For lattice-based Cryptomania, where public-key encryption exists, the main assumption is different. Here, we work with the learning with errors (LWE) problem [Reg09]. Given a random $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a vector of noisy inner products $\mathbf{b} = \mathbf{A}^t\mathbf{s} + \mathbf{e} \bmod q$ for random $\mathbf{s} \in \mathbb{Z}_q^n$ and short $\mathbf{e} \in \mathbb{Z}^m$, the task is to recover $\mathbf{s}$. It admits a search-decision equivalence. Hence, an equivalent problem is to distinguish such $\mathbf{b}$ from uniformly random vectors. The main contributions in this area are encryption schemes [Reg09, KTX07, Pei09, LPR10]. Refer to [MR08, Reg10] for an overview. In addition, there are more advanced constructions, such as identity-based encryption [GPV08, CHKP10, ABB10a, ABB10b] or homomorphic encryption [GHV10].

We build upon the above works and take the next step of adding application-specific features to signature schemes.

1.2 Summary of Results

After introducing the notation, basic definitions, and tools for this thesis in Chapter 2, we spend a chapter on discussing the practical hardness of the two main average-case problems, SIS and LWE. The remaining chapters 4–8 are essentially devoted to constructing lattice-based signature schemes with additional features, while some also contain results of independent interest.

In the following, we summarize the main results in this thesis and defer the discussion of open problems and future research directions to the individual chapters where they appear. We use the "soft-O" notation, where writing $\widetilde{\mathcal{O}}(n)$ means less than $c_1 n \log^{c_2}(n)$ for certain constants c_1, c_2, n_0 and $n \geq n_0$.

Chapter 3: Estimating the Hardness of lattices problems Although there have been many important results and breakthroughs in lattice cryptography, the questions of how to systematically evaluate their security in practice and how to choose secure parameters are still open. This is mainly due to the fact that most security proofs are essentially asymptotic statements. In addition, the hardness of the underlying complexity assumption is controlled by several interdependent parameters rather than just a simple bit length as in many classic schemes.

With our work, we close this gap by providing a framework that (1) distills a hardness estimate out of a given parameter set and (2) relates the complexity of practical lattice-based attacks to symmetric "bit security" for the first time. Our

approach takes various security levels, or attacker types, into account. Moreover, we use it to predict long-term security in a similar fashion as the results that are collected on `www.keylength.com`.

Our framework can be applied in two ways: Firstly, to assess the hardness of proposed parameter sets and secondly, to propose secure parameters in the first place. Our methodology is applicable to essentially all lattice-based schemes that are based on SIS and LWE and it allows us to compare efficiency and security across different schemes and even across different types of cryptographic primitives.

Chapter 4: Blind Signatures Blind signatures, introduced by Chaum [Cha82], have become an important tool in privacy-oriented cryptography.

Generally speaking, such schemes allow a signer to sign a message without seeing it, while retaining a certain amount of control over the process. In particular, the signer can control the number of issued signatures (unforgeability). For the receiver of the signature, this process provides perfect anonymity (blindness), e.g., his spendings remain anonymous when using blind signatures for electronic money.

We propose the first construction from lattices, building upon Lyubashevsky's identification scheme [Lyu08a]. Our scheme offers quasi-linear complexity, statistical blindness, and its unforgeability is based on the hardness of solving SIS for vectors of length $\widetilde{\mathcal{O}}(n^4)$. Using Chapter 3, we propose parameter sets for various security levels to demonstrate that our scheme is quite competitive.

Moreover, it is the first blind signature scheme that supports leakage-resilience, tolerating leakage of a $(1-o(1))$ fraction of the secret key in a model that is inspired by Katz and Vaikuntanathan [KV09].

Chapter 5: Verifiably Encrypted Signatures In a verifiably encrypted signature scheme, signers encrypt their signature under the public key of a trusted third party and prove that they did so correctly. They are a cost-effective tool for fair online contract signing — an important ingredient for today's business processes. The security requirements, due to Boneh et al. [BGLS03], are unforgeability and opacity. Unforgeability prevents malicious users to create verifiably encrypted signatures on behalf of another party and opacity prevents the extraction of ordinary signatures from verifiably encrypted signatures.

We show that the original security model in [BGLS03] is insufficient and propose two novel requirements, which we call *extractability* and *non-frameability*. Extractability ensures that the trusted third party is always able to extract a valid signature from a valid verifiably encrypted signature and non-frameability guaran-

tees that a malicious signer, who cooperates with the trusted party, is not able to forge verifiably encrypted signatures.

The second part of the chapter is devoted to a framework for modular instantiations without inefficient non-interactive zero knowledge proofs to achieve verifiability. We propose two generic constructions based on Merkle authentication trees [Mer89] in the standard model. Furthermore, we extend the specification for verifiably encrypted signatures, bringing them closer to real-world needs. We also argue that the limited capacity of Merkle trees can be a feature in certain business scenarios.

For instance, our framework can be entirely instantiated with lattice-based primitives in the standard model. Not only do we acquire the first verifiably encrypted signature scheme from lattices but also the first efficient pairing-free instantiation in the standard model. The said instantiation is secure based on the hardness of SIS with norm bound $\widetilde{\mathcal{O}}(n^2)$.

Chapter 6: Single-signer Aggregate Signatures Aggregate signatures [BGLS03] mainly serve the purpose of "compressing" signatures and, therefore, saving bandwidth. This is desirable when transmitting multiple signatures in a network with high communication cost (e.g. sensor networks). Since lattice-based signatures are typically large, there is an intrinsic desire to compress them, even regardless of other non-functional constraints.

We put forward the notion of a single-signer aggregate signature as a restricted form of [BGLS03]. Given a list of signatures under the same public key, our scheme allows anyone to compress them into a single one. We introduce a security model and discuss potential applications. Afterwards, we propose a lattice-based construction in the random oracle model. When aggregating ℓ signatures, the compression factor of roughly $\ell/\log(\ell)$ is quite remarkable. Our scheme also supports an interesting extension, namely hierarchical aggregation, which allows the aggregation of aggregates.

Let ℓ_{max} be the maximum number of signatures in an aggregate. We prove aggregate-unforgeability under the assumption that SIS for vectors of length $\ell_{max}\widetilde{\mathcal{O}}(n)$ is hard. Obviously, we need increasingly stronger assumptions with increasing ℓ_{max}. To compensate for this, we apply Chapter 3 to generate parameter sets for various ℓ_{max} with a steady security level and still find the resulting *net* savings to be intriguing.

Chapter 7: Strongly-unforgeable Signatures We propose a variant of the *existentially unforgeable* "bonsai tree" signature scheme [CHKP10]. Our construction offers

the same efficiency as the "bonsai tree" scheme but supports the stronger notion of *strong unforgeability*.

Existential unforgeability merely requires that it is hard to forge a signature for a message that has not been queried to a signature oracle before. In contrast, strong unforgeability also demands that it is hard to re-randomize a given signature, making message-signature pairs somewhat unique.

With our work, we close a gap in the range of available lattice-based signature schemes and provide the first stateless (treeless) signature scheme that supports strong unforgeability in the standard model. Let λ be the output length of a collision-resistant hash function. Then, our scheme is unforgeable if solving SIS for vectors of length $\widetilde{\mathcal{O}}(n\sqrt{\lambda})$ is hard.

Chapter 8: Identity-based Identification In the final chapter of this thesis, we make a brief digression into the field of identity-based identification because signature and identification schemes bear a strong similarity. Identity-based constructions were proposed by Shamir [Sha84] in an attempt to obliterate public-key infrastructures. Instead, such schemes require a key extraction authority that generates the secrets keys for all users under a master public key.

We propose an adaptive-identity secure identity-based identification scheme from lattices, which is secure against malicious verifiers and allows the adaptive corruption of provers. It is secure as long as SIS with norm bounds $\widetilde{\mathcal{O}}(\lambda n^2\sqrt{n})$ is secure, with λ being the output length of a collision-resistant hash function.

Our scheme uses an ideal-lattice interpretation of the "bonsai tree" concept devised in [CHKP10], which we call *convoluted*. It allows us to build an identity-based identification scheme in a new "static identity" model that is weaker than the standard "adaptive identity" model. The static identity model forces the adversary to output a list of to-be-corrupted prover identities before seeing the master public key. Using a re-interpretation of the "hash-sign-switch" paradigm [KR00], we use chameleon hash functions convert these weaker schemes into strong ones.

While there are alternative construction principles, such as the "certification approach" [BNN09], we obtain the first direct construction from lattices and our techniques are transferable to Chapter 4 and Chapter 7. When combined, we achieve identity-based blind signature as well as identity-based strongly-unforgeable signatures in the standard model.

1.3 Relation to Post-Quantum Cryptography

In short, the main goal of "post-quantum cryptography", as opposed to quantum cryptography, is *defeating quantum computers without quantum computers*.

Lattice problems, such as the SIS problem, are conjectured to be immune to quantum computer attacks, which threaten the current security landscape due to Shor's algorithm [Sho97]. Other alternatives for the post-quantum era can be found in the theory of error correcting codes or that of solving non-linear multivariate equation systems. Refer to [BBD08] for an overview.

When confronted with quantum adversaries, however, the common idea of simply exchanging, say, the factoring problem for the SIS problem may fall far too short. Typically, a cryptographic scheme entails much more than just an assumption. For instance, it is accompanied by a proof that may not hold in the modified quantum setting. The most prominent collapsing proof technique would certainly be the Fiat-Shamir paradigm [FS86], which refutes the no-cloning theorem [WZ82]. See, e.g., [FP96, Wat09, DFLS10], for further examples.

In consequence, we do not claim all of the following results to be post-quantum. They are certainly lattice-based with the corresponding benefits attached, and we hope they will pave the way for full-blown post-quantum constructions.

1.4 Conclusion and Open Research Questions

We have demonstrated the great versatility of lattices in cryptographic constructions that do much more than providing ordinary signature or encryption functionality. In essentially all of them, we have seen that former construction principles from the areas of factoring or discrete logarithms cannot be applied directly. In particular, the concept of having objects equipped with a norm causes major complications. We have shown how to overcome some of them.

Judging from past developments, we believe that our constructions will, as building blocks, lead to even more complex schemes in an effort to provide a comprehensive cryptographic landscape from lattices.

While it is unlikely for these constructions to become as strikingly simple and elegant as many designs from, say, pairings, we believe that the potential of ideal lattices has not been fully exploited yet. There are hardly any constructions that use the richer structure in ideal lattices for more than plain efficiency reasons.

In the individual chapters, we have discussed further research directions in the respective area of research. When speaking about lattice cryptography in general,

one of the most pressing issues is the reduction of the size of trapdoor bases. An improvement in this area will immediately affect a large number of derived schemes.

Moreover, a large number of construction principles rely on the existence of trapdoor *permutations* and not all of them can be generalized to use non-bijective trapdoor functions, such as the preimage sampleable functions of [GPV08]. Hence, an efficient construction of such a permutation would be an asset.

Chapter 2

Notation, Definitions & Basic Tools

In this preliminary chapter, we define the shared notation for all subsequent chapters. In addition, we repeat a number of standard security notions and basic tools for self-containment and the reader's convenience.

2.1 General Notation

With n, we always denote the main security parameter. The statement $x \leftarrow_{\$} X$ means that x is chosen uniformly at random from the finite set X. When selecting x from a certain distribution Ψ over X, we write $x \leftarrow_{\Psi} X$ instead. Unless specified otherwise, $\log(\cdot) = \log_2(\cdot)$. We denote $\{1, \ldots, k\}$ by $[k]$ and the (ordered) set $\{x_1, \ldots, x_k\}$ $((x_1, \ldots, x_k))$ by $\{x_i\}_{i=1}^{k}$ $((x_i)_{i=1}^{k})$. Furthermore, we identify $\mathbb{Z}_q$ with $\{-\lceil q/2 \rceil + i\}_{i=1}^{q}$ and $[a, b]_{\mathbb{Z}} := [a, b] \cap \mathbb{Z}$. For the disambiguation of assignment and comparison, we use "$\leftarrow$" for assignment and "$=$" as well as "$\equiv$" for comparison. When using "$\equiv$", we mean equality modulo an equivalence relation, e.g., $a \equiv b \pmod{q}$. Occasionally, we omit the relation and write $a \equiv b$ when the meaning is clear from the context. When we write "$\|$", we mean the concatenation of strings, vector entries, or matrix columns, depending on the context. Consequently, we define the prefix relation $\sqsubset$ for these objects: we write $a \sqsubset b$ if there is a c such that $b = a\|c$; the negation is $a \not\sqsubset b$. For a string $s \in \{0,1\}^*$, we denote its bit length with $|s|$. Occasionally, we use "iff" as shorthand for "if and only if".

2.2 Algorithms

Algorithms are denoted in sans-serif font, e.g., DoIt. When the algorithm is treated as a black box, e.g., an adversary in a reduction proof, we use calligraphy as in $\mathcal{A}$. The joint execution of two algorithms $\mathcal{A}$ and $\mathcal{B}$ in an interactive protocol with private inputs x to $\mathcal{A}$ and y to $\mathcal{B}$ is written as $(a, b) \leftarrow \langle \mathcal{A}(x), \mathcal{B}(y) \rangle$. The private outputs are a for $\mathcal{A}$ and b for $\mathcal{B}$. When writing $\langle \mathcal{A}(x), \mathcal{B}(y) \rangle^{k}$, we stress that the interaction can take place up to $k \in \mathbb{N} \cup \{\infty\}$ times.

Algorithms are *efficient* if they run in probabilistic polynomial time (PPT) $\mathrm{poly}(n) := n^k$, for any positive real constant k. To stress that they run in polynomial time in a certain parameter p, we use the encoding 1^p instead. Algorithms are *sub-exponential* (SUBEXP) if they require at most $2^{o(n)}$ computational steps. If an algorithm $\mathcal{A}(x)$ is probabilistic, it has access to a perfect random bit generator while processing the input x. Alternatively, we write $\mathcal{A}(x; \rho)$ to provide $\mathcal{A}$ with a random tape ρ to make it deterministic.

Let X, Y be two computational problems. The relation $\mathrm{X} \leq \mathrm{Y}$ means that the problem Y is at least as hard to solve as X and that there is a reduction algorithm $\mathcal{R}$ that reduces X to Y. Typically, $\mathcal{R}$ is a PPT algorithm and occasionally, we only require $\mathcal{R}$ to be in SUBEXP.

To estimate running time and object sizes, we use the following standard Landau notation for asymptotic growth.

Growth	**Condition**
$f(n) = \mathcal{O}(g(n))$	$\limsup_{n\to\infty} \left\lvert \frac{f(n)}{g(n)} \right\rvert < \infty$
$f(n) = o(g(n))$	$\lim_{n\to\infty} \left\lvert \frac{f(n)}{g(n)} \right\rvert = 0$
$f(n) = \Omega(g(n))$	$0 < \liminf_{n\to\infty} \left\lvert \frac{f(n)}{g(n)} \right\rvert$
$f(n) = \omega(g(n))$	$\lim_{n\to\infty} \left\lvert \frac{f(n)}{g(n)} \right\rvert = \infty$
$f(n) = \Theta(g(n))$	$0 < \liminf_{n\to\infty} \left\lvert \frac{f(n)}{g(n)} \right\rvert$
	$\leq \limsup_{n\to\infty} \left\lvert \frac{f(n)}{g(n)} \right\rvert < \infty$

In addition, we use the "soft-$\mathcal{O}$"-notation, e.g., $\widetilde{\mathcal{O}}$ or $\widetilde{\Omega}$, that neglect and hide poly-logarithmic factors $\mathrm{polylog}(n) := \mathrm{poly}(\log(n))$.

2.3 Cryptographic Primitives

Since they are needed in various parts of this theses, we recap the definitions and security models for trapdoor one-way functions, collision-resistant hash functions, chameleon hash functions, as well as digital signature and encryption schemes.

Notational Convention Whenever a primitive has a public key pk and a private key sk, we assume that the secret key contains the public key and omit pk from the list of arguments for the algorithms in the primitive.

A function $f(n)$ is negligible if it vanishes faster than $1/g(n)$ for any polynomial g. If f is not negligible, we use the terms *non-negligible* or *noticeable*. Based on this definition, a probability $p = p(n)$ can be (non-)negligible and it is *overwhelming* if $1 - p(n)$ is negligible.

When talking about the success probability of an algorithms $\mathcal{A}$, we mean that the probability is taken over the coin tosses in the environment of $\mathcal{A}$ as well as over $\mathcal{A}$'s internal randomness.

2.3.1 Collision-resistant Hash Functions

A family $\mathcal{H}$ of hash functions $\mathsf{H} : \{0,1\}^* \to \{0,1\}^\lambda$, $\lambda = \lambda(n)$, is collision-resistant if there is no efficient adversary $\mathcal{A}$ that wins in the experiment $\mathsf{Exp}^{\mathsf{CR}}_{\mathcal{A},\mathcal{H}}$ with non-negligible probability. On input a description H of a randomly chosen function from $\mathcal{H}$, the task is to find distinct strings x and x' such that they collide under H.

Experiment $\mathsf{Exp}^{\mathsf{CR}}_{\mathcal{A},\mathcal{H}}(n)$
$\mathsf{H} \leftarrow_{\$} \mathcal{H}(n)$
$(x, x') \leftarrow \mathcal{A}(\mathsf{H})$
Return 1 iff $\mathsf{H}(x) = \mathsf{H}(x')$ and $x \neq x'$.

Occasionally, for ease of presentation, we refer to a collision-resistant hash function instead of to the corresponding family.

2.3.2 Trapdoor One-way Functions

Trapdoor one-way functions are one-way functions that admit a secret key to selectively lift one-wayness for the secret-key holder.

A family $\mathrm{TFF} = (\mathsf{Kg}, \mathsf{Eval}, \mathsf{Inv})$ of trapdoor one-way functions can be specified as follows.

Key Generation: $\mathsf{Kg}(1^n)$ outputs a private key sk and a public key pk.

Evaluation: $\mathsf{Eval}(pk, x)$ outputs an image $y \in R$ in the range R under pk, for an input $x \in D$ from the domain D.

Inversion: $\mathsf{Inv}(sk, y)$ outputs $x' \in D$ such that $\mathsf{Eval}(pk, x') = y$ if $y \in \{\mathsf{Eval}(pk, x) : x \in D\}$; and $\perp$ otherwise.

As for correctness, we require that for all honestly generated keys and all honestly generated y, the above specification yields $x = x'$ (with overwhelming probability).

Security, i.e., trapdoor one-wayness, is defined in the experiment $\mathsf{Exp}^{\mathsf{TRAP\text{-}OW}}_{\mathcal{A},\mathrm{TFF}}$, where the adversary $\mathcal{A}$ receives a public key and the image y for a random element

x of the domain. $\mathcal{A}$ wins the game if it can recover x with noticeable probability.

Experiment $\mathsf{Exp}^{\mathsf{TRAP\text{-}OW}}_{\mathcal{A},\mathrm{TFF}}(n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$x \leftarrow_{\$} D$
$y \leftarrow \mathsf{Eval}(pk, x)$
$x' \leftarrow \mathcal{A}(pk, y)$
Return 1 iff $x = x'$.

2.3.3 Chameleon Hash Functions

Krawczyk and Rabin [KR00] put forward the notion of chameleon hash functions, i.e., collision-resistant hash functions with a trapdoor. We use a slight generalization of their definition. A family $\mathcal{C} = (\mathsf{Kg}, \Psi)$ of chameleon hash functions has the following specification.

Key Generation: $\mathsf{Kg}(1^n)$ outputs a pair $(\mathsf{C}, \mathsf{C}^{-1})$, with C being the public function for evaluation and C^{-1} being the private algorithm for inversion.

Randomness: The randomness $\rho \in \mathcal{R}$ is chosen from the efficiently sampleable distribution Ψ over $\mathcal{R}$, i.e., $\rho \leftarrow_{\Psi} \mathcal{R}$.

Evaluation: $\mathsf{C}(msg, \rho)$, on input a message $msg \in \mathcal{M}$ from the message space $\mathcal{M}$ and a random value $\rho \in \mathcal{R}$, outputs an image $\mu \in \mathcal{I}$. The pair (C, μ) is distributed statistically close to uniform.

Inversion: $\mathsf{C}^{-1}(\mu, msg)$, on input a target image $\mu \in \mathcal{I}$ and a source message $msg \in \mathcal{M}$, samples and outputs $\rho \in \mathcal{R}$ such that $\mathsf{C}(msg, \rho) = \mu$.

Collision resistance of $\mathcal{C}$ is defined as for ordinary hash functions, namely no efficient algorithm can find $(msg, \rho) \neq (msg', \rho)$ with $\mathsf{C}(msg, \rho) = \mathsf{C}(msg', \rho)$ with noticeable probability. The interesting property of chameleon hash functions is that the following two processes yield statistically indistinguishable distributions for any $(\mathsf{C}, \mathsf{C}^{-1}) \leftarrow \mathsf{Kg}(1^n)$ and for any given document $msg \in \mathcal{M}$.

Forward: Pick $\rho \leftarrow_{\Psi} \mathcal{R}$ and set $\mu \leftarrow \mathsf{C}(msg, \rho)$. The output is $X \leftarrow (msg, \rho, \mu)$.

Backward: Pick a $\mu \leftarrow_{\$} \mathcal{I}$ and set $\rho \leftarrow \mathsf{C}^{-1}(\mu, msg)$. The output is $Y \leftarrow (msg, \rho, \mu)$.

Recall that the statistical distance of two random variables X, Y over a discrete domain D is defined as $\Delta(X,Y) = 1/2 \sum_{a \in D} |\operatorname{Prob}[X=a] - \operatorname{Prob}[Y=a]|$. Hence, we require that

$$\Delta(X,Y) = 1/2 \sum_{(a,b,c) \in (\mathcal{M},\mathcal{R},\mathcal{I})} |\operatorname{Prob}[X=(a,b,c)] - \operatorname{Prob}[Y=(a,b,c)]|$$

is negligible.

Chameleon hash functions $\mathsf{C} : \mathcal{M} \times \mathcal{R} \to \mathcal{I}$ can be composed with regular hash functions $\mathsf{H} : \{0,1\}^* \to \mathcal{M}$, with the result being another chameleon hash function.

2.3.4 Signature Schemes

Digital signature schemes $\mathrm{DS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ are specified as follows.

Key Generation: $\mathsf{Kg}(1^n)$ outputs a private signing key sk and a public verification key pk.

Signing: $\mathsf{Sign}(sk, msg)$ outputs a signature σ under sk for a message msg from the message space $\mathcal{M}$.

Verification: $\mathsf{Vf}(pk, \sigma, msg)$ outputs 1 if and only if σ is a valid signature for msg under pk.

Correctness is defined in a straightforward way, namely that every honestly generated signature under honestly generated keys should be valid (with overwhelming probability).

Typically, signature schemes are proven to be existentially unforgeable under chosen message attacks (EU-CMA) [GMR88], but we will also consider the stronger notion of *strong* unforgeability under chosen message attacks (SU-CMA). Both notions are defined through a game, or experiment, with an adversary $\mathcal{A}$. For, EU-CMA, the following experiment $\mathsf{Exp}^{\mathsf{EU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}$ gives $\mathcal{A}$ access to the public key pk and to a signature oracle OSign, which $\mathcal{A}$ can query adaptively. The adversary wins the game if it outputs a valid message signature pair (msg^*, σ^*) such that msg^* has not been queried to OSign before.

Experiment $\mathsf{Exp}^{\mathsf{EU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}(n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$(msg^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{OSign}(sk,\cdot)}(pk)$
Let $((msg_i, \sigma_i))_{i=1}^{Q_{\mathsf{OSign}}}$ be the query-answer pairs of $\mathsf{OSign}(sk, \cdot)$.
Return 1 iff $\mathsf{Vf}(pk, \sigma^*, msg^*) = 1$ and $msg^* \notin (msg_i)_{i=1}^{Q_{\mathsf{OSign}}}$.

Strong unforgeability is defined in a similar game $\mathsf{Exp}^{\mathsf{SU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}$, without artificially restricting the space of valid forgeries as in the last line of $\mathsf{Exp}^{\mathsf{EU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}$. Now, the adversary already wins if it outputs a signature σ^* that has never been returned by OSign.

Experiment $\mathsf{Exp}^{\mathsf{SU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}(n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$(msg^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{OSign}(sk,\cdot)}(pk)$
Let $((msg_i, \sigma_i))_{i=1}^{Q_{\mathsf{OSign}}}$ be the query-answer pairs of $\mathsf{OSign}(sk, \cdot)$.
Return 1 iff $\mathsf{Vf}(pk, \sigma^*, msg^*) = 1$ and $(msg^*, \sigma^*) \notin ((msg_i, \sigma_i))_{i=1}^{Q_{\mathsf{OSign}}}$.

A scheme DS is existentially (strongly) unforgeable if no efficient adversary has a non-negligible success probability in the respective experiment. We can also make this statement more precise by saying that DS is $(t, Q_{\mathsf{OSign}}, \epsilon)$-existentially (strongly) unforgeable if there is no adversary, running in time t, which succeeds with probability more than ϵ after making at most Q_{OSign} signature oracle queries. Similar definitions apply to one-time signature schemes OTS $= (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$, where $\mathcal{A}$ is only allowed to query a single message to the signature oracle.

2.3.5 Encryption Schemes

A public key encryption scheme PKE $= (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ has the following specification.

Key Generation: $\mathsf{Kg}(1^n)$ outputs a private decryption key sk and a public encryption key pk.

Encryption: $\mathsf{Enc}(pk, msg)$ outputs a ciphertext ct under pk, for a message msg from the message space $\mathcal{M}$.

Decryption: $\mathsf{Dec}(sk, ct)$ outputs $msg' \in \mathcal{M}$ if ct is a valid ciphertext and $\perp$ otherwise.

For correctness, we require that for all honestly generated keys and all honest generated ciphertexts, the above specification yields $msg = msg'$ (with overwhelming probability).

PKE is indistinguishable under chosen plaintext attacks (IND-CPA) if no efficient algorithm $\mathcal{A}$ can determine which one of two chosen plaintexts has been encrypted in a given ciphertext [GM84]. The corresponding experiment $\mathsf{Exp}^{\mathsf{CPA}}_{\mathcal{A},\mathrm{PKE}}$ gives $\mathcal{A}$ access

to an encryption oracle $\mathsf{OEnc}(pk, b, \cdot, \cdot)$, where $b \in \{0,1\}$ is kept outside the view of $\mathcal{A}$. $\mathsf{OEnc}(pk, b, \cdot, \cdot)$ takes as input two messages msg_0 and msg_1 with $|msg_0| = |msg_1|$ and returns $\mathsf{Enc}(pk, msg_b)$. The adversary wins if it is able to guess b with probability noticeably greater than $1/2$. The stronger notion of a chosen ciphertext attack (CCA) is only of marginal interest in this thesis; refer to [BDPR98] for an overview.

Experiment $\mathsf{Exp}^{\mathsf{CPA}}_{\mathcal{A},\mathrm{PKE}}(n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$b \leftarrow_{\$} \{0,1\}$
$d \leftarrow \mathcal{A}^{\mathsf{OEnc}(pk,b,\cdot,\cdot)}(pk)$
Return 1 if and only if $d = b$.

We say that PKE is CPA secure if there is no efficient adversary $\mathcal{A}$ that has a non-negligible distinguishing advantage $\mathsf{Adv}^{\mathsf{CPA}}_{\mathcal{A},\mathrm{PKE}}$, where

$$\mathsf{Adv}^{\mathsf{CPA}}_{\mathcal{A},\mathrm{PKE}} := \left| \mathrm{Prob}\left[\mathsf{Exp}^{\mathsf{CPA}}_{\mathcal{A},\mathrm{PKE}}(n) = 1 \right] - 1/2 \right| .$$

2.4 The Random Oracle Model

In the *random oracle* model [BR93], as opposed to the *standard* or *plain* model, a family of hash functions $\mathsf{H} : \{0,1\}^* \to \{0,1\}^{\lambda(n)}$ can be modeled as a family of truly random functions. It is assumed that all algorithms have black-box access to H, i.e., they do not know how it is implemented. Upon a new query x, the random oracle is supposed to answer with a uniformly random sample $y \leftarrow_{\$} \{0,1\}^{\lambda(n)}$. From this point on, it has to respond consistently, i.e., $\mathsf{H}(x)$ always returns y.

When used in reduction proofs, the random oracle allows the reduction algorithm to adaptively program the input-output behavior outside of the view of the remaining algorithms. This technique allows security proofs for schemes that are otherwise hard or impossible to prove secure under standard assumptions. A popular example is the (full-domain hash) RSA signature scheme [BR96]. On the negative side, random oracles do not exist in the "real" world and they are, rigorously speaking, a false assumption. In the real world, one has to rely on actual hash functions that have a polynomial-size program. Furthermore, there are (artificial) counter-examples that become insecure whenever the random oracle is replaced with a real function [CGH04]. Hence, the entire concept is viewed as controversial and eliminating the need for random oracles is a goal in its own right. Nevertheless, we will use the random oracle model whenever it seems unavoidable and accept that the resulting proofs are heuristic.

Relation to Post-quantum Cryptography In addition to the dispute in the classical setting, the presence of quantum adversaries causes further complications. First, the security of hash functions in general degrades due to improved collision-search algorithms [Gro96, BHT98]. Second, whenever the random oracle is used to extract knowledge from the adversary, e.g., in the Fiat-Shamir paradigm [FS86], the random oracle is adaptively and lazily re-programmed and the adversary is often rewound to an earlier (quantum) state. The former may be impossible if the adversary is allowed to query superpositions to the random oracle and the latter is forbidden by the no-cloning theorem [WZ82]. Furthermore, the reduction may, through its actions cause slight, yet noticeable disturbances in the quantum environment [FP96]. Potential solutions are discussed in [Wat09] and we refer the interested reader to [DFLS10] for a cryptographers' view of these issues.

2.5 Witness-indistinguishable Proofs of Knowledge

Let R be an $\mathcal{NP}$ relation $\{(x,w)\} \subseteq \{0,1\}^{l(n)} \times \{0,1\}^{l(n)}$ for $l(n) = \text{poly}(n)$. Then, there is an efficient algorithm that, given (x,w) decides membership in R. The associated language is $L_R = \{x \in \{0,1\}^{l(n)} : \exists w \in \{0,1\}^{l(n)} \text{ such that } (x,w) \in R\}$. If $(x,w) \in R$, we call w a *witness* for x being in L_R. With $W(x)$, we denote the witness-set $\{y \in \{0,1\}^{l(n)} : (x,y) \in R\}$.

In a *proof of knowledge*, a two-party proof system with a PPT prover $\mathcal{P}$ and a PPT verifier $\mathcal{V}$, $\mathcal{P}(x,w)$ convinces $\mathcal{V}(x,a)$ that it knows a witness $w \in W(x)$. We write $b \leftarrow \langle \mathcal{P}, \mathcal{V} \rangle ((x,w),(x,a))$, where $b \in \{0,1\}$ indicates whether $\mathcal{V}$ accepts the proof or not. The string a is auxiliary information or a prefix of the conversation.

Let $p(n)$ be the probability for $b = 1$ (accept) and let $e(n)$ be the "knowledge error", i.e., the probability that $\mathcal{V}$ falsely accepts. We demand that $p(n)$ is positive for honest provers and that cheating provers cannot efficiently convince honest verifiers without knowing a witness, but with probability negligibly close to $e(n)$. The proof systems admits a *knowledge extractor*, which can extract $w \in W(x)$ from a (cheating) prover $\mathcal{P}$ for $x \in L_R$ in expected time proportional to $1/(p(n)) - e(n))$ via black-box access.

Let $W(x) > 1$ for all x. The proof system is *statistically witness indistinguishable* if $\mathcal{V}$ cannot efficiently distinguish between any two witnesses $w_1, w_2 \in W(x)$. Hence, we require for all $\mathcal{V}$ that the statistical distance

$$\Delta(\langle \mathcal{P}, \mathcal{V} \rangle ((x,w_1),(x,a)), \langle \mathcal{P}, \mathcal{V} \rangle ((x,w_2),(x,a)))$$

is negligible.

We refer the reader to, e.g., [FS90, BG92], for a comprehensive discussion. In this work, we require the fact the witness-indistinguishable proofs hide the witness from the verifier and that they can be securely composed in parallel.

2.6 Cryptographic Lattices

In this work, our main concern is with full-dimension, or full-rank, lattices of a special form. A (full-dimensional) lattice in $\mathbb{R}^d$ is a discrete subgroup $\Lambda = \{\sum_{i=1}^{d} x_i \mathbf{b}_i : x_i \in \mathbb{Z}\}$, typically represented by a *basis* matrix $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_d] \in \mathbb{Z}^{d \times d}$ of $\mathbb{R}$-linearly independent vectors. We write $\Lambda = \Lambda(\mathbf{B})$ when Λ is generated by $\mathbf{B}$. The number d of linearly independent vectors in any such basis is the *dimension* $\dim(\Lambda)$ of a lattice. For a lattice $\Lambda = \Lambda(\mathbf{B})$, its *dual lattice* Λ^* is the set of all $\mathbf{x} \in \mathbb{R}^d$ with $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}$ for all $\mathbf{y} \in \Lambda$. In dimension $d > 1$, a given lattice Λ has infinitely many bases. A quality metric for bases is the basis length $\|\mathbf{B}\| := \max_{i \in [d]}\{\|\mathbf{b}_i\|_2\}$. Given any basis $\mathbf{B}$ of the lattice Λ, the *determinant* $\det(\Lambda)$ of the lattice is $\sqrt{\det(\mathbf{B}^t\mathbf{B})}$. Both, dimension and determinant are invariants of the lattice. Another set of invariants are the successive minima. The i-th *successive minimum* $\lambda_i(\Lambda)$ is the smallest radius of a sphere that contains i linearly independent vectors in Λ.

In cryptography, we use lattices of a special form, which we call *q-ary*. Let $n \in \mathbb{N}$, $q \in \mathbb{N}$, $m \in \mathbb{N}$, and $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, then the associated q-ary lattice is $\Lambda_q^{\perp}(\mathbf{A}) := \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{A}\,\mathbf{v} \equiv \mathbf{0} \pmod{q}\}$. Its, up to scaling, dual lattice is $\Lambda_q(\mathbf{A}) := \{\mathbf{w} \in \mathbb{Z}^m : \exists \mathbf{e} \in \mathbb{Z}_q^n \text{ s.t. } \mathbf{A}^t\mathbf{e} \equiv \mathbf{w} \pmod{q}\}$, i.e., we have $1/q \cdot \Lambda_q^{\perp}(\mathbf{A}) = (\Lambda_q(\mathbf{A}))^*$. For $\mathbf{A} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$, prime q, and $m > n$, the determinant of the corresponding q-ary lattice $\Lambda_q^{\perp}(\mathbf{A})$ is q^n with high probability and typically, we have $m = \Omega(n \log(n))$. The matrices $\mathbf{A}$ bear similarities with parity check matrices for error correcting codes.

2.6.1 Lattice Problems

One of the main computational problems in general lattices is the approximate shortest vector problem (SVP). Given a basis $\mathbf{B}$ of Λ and an approximation factor $\gamma \geq 1$, the task is to find a vector $\mathbf{v} \in \Lambda$ with $0 < \|\mathbf{v}\| \leq \gamma\lambda_1(\Lambda)$. A related problem is the approximate shortest independent vector problem (SIVP), where given a basis $\mathbf{B}$ of Λ and an approximation factor γ, the task is to find a set $\{\mathbf{v}_1, \ldots, \mathbf{v}_d\}$ of linearly independent vectors in Λ such that $\max_{i \in [d]}\{\|\mathbf{v}_i\|\} \leq \gamma\lambda_d$. We say that an algorithm solves one of the above problems in dimension d if it can solve every instance, i.e., in the worst case. The best known algorithms for solving them with $\gamma \leq \mathrm{poly}(d)$ require exponential time [Kan83], or time and space [MV10a, MV10b].

In lattice-based cryptography, however, we build upon specific average-case problems. The main computational problem in a q-ary lattice $\Lambda_q^{\perp}(\mathbf{A})$ is the following short integer solution problem (SIS). Given an instance $\mathbf{A} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$ for parameters (n, m, q, ν), the task is to find $\mathbf{v} \in \Lambda_q^{\perp}(\mathbf{A})$ with $0 < \|\mathbf{v}\| \leq \nu$ for a given polynomial norm bound $\nu > 0$. Basically, the SIS problems was introduced and analyzed in Ajtai's seminal work [Ajt96] but there are numerous improvements of the analysis, e.g., [MR07, GPV08].

For $\Lambda_q(\mathbf{A})$, the dual lattice, we consider the learning with errors problem (LWE) as the main computational problem. Its parameters are (n, m, q) as before and a (truncated) probability distribution Ψ over $\mathbb{Z}^m$ with finite support. An instance of the *search* version consists of $\mathbf{A} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$ as well as m "noisy" inner products $\mathbf{b} \leftarrow \mathbf{A}^t\mathbf{s} + \mathbf{e} \bmod q$ with $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$ and $\mathbf{e} \leftarrow_{\Psi} \mathbb{Z}^m$. The task is to recover $\mathbf{s}$. The *decision* version of LWE is defined as follows. Flip a coin $c \leftarrow_{\$} \{0, 1\}$; if $c = 0$, let $\mathbf{b} \leftarrow_{\$} \mathbb{Z}_q^m$; otherwise compute $\mathbf{b} \leftarrow \mathbf{A}^t\mathbf{s} + \mathbf{e} \bmod q$ for $\mathbf{s} \leftarrow_{\$} \mathbb{Z}_q^n$ and $\mathbf{e} \leftarrow_{\Psi} \mathbb{Z}^m$. Given $(\mathbf{A}, \mathbf{b})$, the task is to determine c with success probability more than $1/2$. Both versions are defined and analyzed in Regev's groundbreaking work [Reg09]. There, he also shows that search and decision LWE are essentially equivalent for appropriate parameters. The "standard" noise distribution Ψ_α^m is a component-wise m-dimensional discretized Gaussian distribution over $\mathbb{Z}^m$ with parameter $\alpha = \alpha(n)$. It is spherical with standard deviation $q\alpha/\sqrt{2\pi}$.

Notice that we have not specified the norm for any of the above lattice problems. This is because they can be defined for arbitrary norms. If not explicitly stated, e.g., SIVP$^{\infty}$ for the infinity (ℓ_∞) norm, we work in the Euclidean (ℓ_2) norm.

2.6.2 Worst-case to Average-case Connection

In [Ajt96], Ajtai describes a reduction that connects the average-case problems SIS with the worst-case problem SIVP. Basically, it states that *any* instance of SIVP in dimension n can be phrased as a *random* instance of SIS. Hence, solving a non-negligible portion of instances of SIS in PPT (SUBEXP) yields a PPT (SUBEXP) algorithm that can solve all instances of SIVP. This worst-case to average-case reduction has been improved in, e.g., [MR07], and the latest version is due to Gentry, Peikert, and Vaikuntanathan [GPV08].

Proposition 2.1. *For any $\nu \leq \mathrm{poly}(n)$, prime $q \geq \nu g(n)$ for $g(n) = \omega(\sqrt{n \log(n)})$, and $m \geq 2n \log(q) = \Omega(n \log(n))$, the average-case problem SIS(n, m, q, ν) is at least as hard as γ-SIVP in dimension n in the worst case with $\gamma = \nu\, \widetilde{\mathcal{O}}(\sqrt{n})$.*

When introducing LWE, Regev also proves a worst-case to average-case reduc-

tion in [Reg09]. For the "standard" noise distribution Ψ_α^m, the associated quantum reduction can be formulated as follows and a similar non-quantum version can be found in [Pei09] at the expense of stronger assumptions or exponential moduli.

Proposition 2.2. *Let $\alpha = \alpha(n)$ with $0 < \alpha < 1$, and prime $q > 2\sqrt{n}/\alpha$, and $m = m(n) \leq \mathrm{poly}(n)$. The average-case search problem LWE(n, m, q) with the standard noise distribution $\Psi = \Psi_\alpha^m$ is at least as hard as quantumly solving γ-SIVP in dimension n in the worst case with $\gamma = \widetilde{\mathcal{O}}(n/\alpha)$.*

2.6.3 Ideal Lattices

Representing a q-ary lattice requires $mn\log(q) = \Omega(n^2\log^2(n))$ bits of storage, which can be a serious disadvantage in practice. In part, this caveat can be lifted by restricting the underlying lattice problems to the subclass of *ideal lattices*.

Let $\mathbf{R}$ be a ring, then $I \subseteq \mathbf{R}$ is an *ideal* in $\mathbf{R}$ if it is an additive subgroup of $\mathbf{R}$ that is closed under multiplication with $\mathbf{R}$. When we say that a lattice corresponds to an ideal, we mean that its vectors can be interpreted, via an embedding, as elements in I and vice versa. Since I is closed under addition and multiplication, we obtain lattices with a richer structure.

We build cryptography upon a particular ring, namely the ring of integer polynomials $\mathbb{Z}[X]$ modulo the ideal $\langle \mathbf{f} \rangle \subset \mathbb{Z}[X]$ generated by a monic, irreducible polynomial $\mathbf{f}$ of degree n. For simplicity and efficiency, we typically fix $\mathbf{f} = X^n + 1$. While this choice simplifies notation and also the analysis of the resulting cryptosystems, essentially all results can be generalized.

Let $\mathbf{R}_0 = \mathbb{Z}[X]/\langle X^n + 1\rangle$ and $\mathbf{R} = \mathbb{Z}_q[X]/\langle X^n + 1\rangle$. Via a coefficient embedding, we have $\mathbf{R}_0 \cong \mathbb{Z}^n$ and $\mathbf{R} \cong \mathbb{Z}_q^n$. Furthermore, we have the modules $\mathbf{R}_0^m \cong \mathbb{Z}^{mn}$ and $\mathbf{R}^m \cong \mathbb{Z}_q^{mn}$. Note that, typically, $m = \mathcal{O}(\log(n))$ in ideal lattices as opposed to $m = \Omega(n\log(n))$ in the q-ary case.

Elements of the modules are denoted with $\hat{\mathbf{a}} = (\mathbf{a}_1, \ldots, \mathbf{a}_m)$, which defines the ideal lattice $\Lambda_{\mathbf{R}}^{\perp}(\hat{\mathbf{a}}) := \{\hat{\mathbf{x}} \in \mathbf{R}_0^m : \hat{\mathbf{a}} \circledast \hat{\mathbf{x}} \equiv \mathbf{0} \in \mathbf{R}\}$. The product "$\circledast$" is a scalar product, i.e., $\hat{\mathbf{a}} \circledast \hat{\mathbf{x}} := \sum_{i=1}^m \mathbf{a}_i\mathbf{x}_i$. For all $\hat{\mathbf{a}}, \hat{\mathbf{b}} \in \mathbf{R}_0$ and for all $\mathbf{r} \in \mathbf{R}_0$, we have $\hat{\mathbf{a}} \circledast \hat{\mathbf{b}} = \hat{\mathbf{b}} \circledast \hat{\mathbf{a}}$ and $\mathbf{r}(\hat{\mathbf{a}} \circledast \hat{\mathbf{b}}) = (\mathbf{r}\hat{\mathbf{a}}) \circledast \hat{\mathbf{b}} = \hat{\mathbf{a}} \circledast (\mathbf{r}\hat{\mathbf{b}})$. When mixing operands from $\mathbf{R}_0$ and $\mathbf{R}$, equality holds modulo q.

Using the definition of an ideal, it is easy to interpret an ideal lattice as a q-ary lattices of special structure. Given $\Lambda_{\mathbf{R}}^{\perp}(\hat{\mathbf{a}})$, we define the corresponding q-ary lattice $\Lambda_q^{\perp}(\mathbf{A})$ via the matrix $\mathbf{A} = \mathbf{A}_1 \| \cdots \| \mathbf{A}_m$. For all $i \in [n]$ and $j \in [m]$, we let the i-th column of $\mathbf{A}_j$ be the embedding of $\mathbf{a}_j X^i$. Both lattices are identical, but $\hat{\mathbf{a}}$ is the more compact representation with only $\mathcal{O}(n\log^2(n))$ bits, while $\mathbf{A}$ requires

$\Omega(n^2 \log^2(n))$ bits.

Since ideal lattices are a subclass of q-ary lattices, the above problem definitions carry over, where the norm of an element in $\mathbf{R}_0$ ($\mathbf{R}_0^m$) is defined as the norm of the respective coefficient embedding into $\mathbb{Z}^n$ ($\mathbb{Z}^{mn}$).

For a more formal treatment, refer to, e.g., [Mic07, LM06], and for the a different (canonical) embedding to [PR07, LPR10].

2.7 Cryptographic Tools from Lattices

Throughout the recent years, a number of tools, based on the worst-case hardness of lattice problems, have been developed. In this theses, we build upon three main tools: collision-resistant compression functions, trapdoors for q-ary lattices, and preimage-sampleable trapdoor functions. They are outlined in the following and we discuss their relation to the standard definitions for cryptographic primitives from Section 2.3.

2.7.1 Collision-resistant Compression Functions

Lattice-based compression functions can be built upon q-ary and ideal lattices. The initial construction in q-ary lattices is due to Ajtai [Ajt96]; see also Goldreich, Goldwasser, and Halevi [GGH96]. The corresponding family of compression functions is keyed with a random matrix $\mathbf{A} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$ and it maps $\mathbb{Z}^m \to \mathbb{Z}_q^n$ via $\mathbf{x} \mapsto \mathbf{A}\mathbf{x} \bmod q$. Finding $\mathbf{x} \neq \mathbf{x}'$ such that $\max(\|x\|, \|x\|') \leq d$ and $\mathbf{A}\mathbf{x} \equiv \mathbf{A}\mathbf{x}'$ immediately yields a solution $\mathbf{x} - \mathbf{x}'$ to $\mathrm{SIS}(n, m, q, 2d)$.

Inspired by Ajtai's construction and Micciancio's one-way function [Mic07], Lyubashevsky and Micciancio introduce the family $\mathcal{H}(\mathbf{R}, m)$ of collision-resistant compression functions in ideal lattices [LM06]. The functions $h \in \mathcal{H}$ map $\mathbf{R}_0^m \to \mathbf{R}$. When the input is restricted to a set $D^m \subseteq \mathbf{R}_0^m$ of small-norm polynomials, the function family is collision-resistant according to Section 2.3.1. The domain can be extended to $\{0,1\}^*$ via standard techniques as described in [LMPR08].

A random element of the family is indexed with $\hat{\mathbf{a}} \leftarrow_{\$} \mathbf{R}^m$ and the associated function $h = h_{\hat{\mathbf{a}}}$ maps $\hat{\mathbf{x}} \in D^m$ to $\hat{\mathbf{a}} \circledast \hat{\mathbf{x}} \bmod q$. Obviously, h is linear in the sense that $h(\mathbf{r}(\hat{\mathbf{x}} + \hat{\mathbf{y}})) \equiv \mathbf{r}(h(\hat{\mathbf{x}}) + h(\hat{\mathbf{y}})) \pmod q$ for all $\mathbf{r} \in \mathbf{R}_0$ and $\hat{\mathbf{x}}, \hat{\mathbf{y}} \in \mathbf{R}_0^m$.

In most of our constructions, we use the following *collision problem* $\mathrm{COL}(\mathcal{H}(\mathbf{R}, m), D)$. It picks $h \leftarrow_{\$} \mathcal{H}(\mathbf{R}, m)$ and asks the adversary to find a distinct pair $(\hat{\mathbf{x}}, \hat{\mathbf{x}}') \in D^m \times D^m$ such that $h(\hat{\mathbf{x}}) = h(\hat{\mathbf{x}}')$. Solving $\mathrm{COL}(\mathcal{H}(\mathbf{R}, m), D)$ with non-negligible probability in PPT (SUBEXP) implies a PPT (SUBEXP) algorithm that solves

SIVP in the worst-case in lattices that correspond to an ideal in $\mathbf{R}_0$ via the following proposition.

Proposition 2.3 (Theorem 2 in [LM06]). *Let $D = \{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d\}$, $m > \log(q)/\log(2d)$, and $q \geq 4dmn\sqrt{n}\log(n)$. An adversary $\mathcal{A}$ that solves the average-case problem $COL(\mathcal{H}(\mathbf{R}, m), D)$ can be used to solve $SIVP^\infty$ with approximation factors $\gamma = d\,\widetilde{\mathcal{O}}(mn)$ in the worst case in lattices that correspond to ideals in $\mathbf{R}_0$.*

2.7.2 Trapdoors for q-ary Lattices

As mentioned in Section 2.6, a lattice $\Lambda_q^\perp(\mathbf{A})$ has many bases. But, only those bases $\mathbf{T}$ with a small norm $\|\mathbf{T}\|$ qualify as trapdoors, which can be used to solve SIS as well as LWE. This concept is one of the cornerstones in lattice-based cryptography as it is used in many signature and encryption schemes.

Improving Ajtai's work [Ajt99], Alwen and Peikert show how to generate a random matrix $\mathbf{A}$ together with a short trapdoor basis $\mathbf{T}$ in [AP09]. In addition, they bound the length of the *Gram-Schmidt orthogonalization (GSO)* of $\mathbf{T}$. Let $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_m]$. Its GSO $\tilde{\mathbf{T}} = [\tilde{\mathbf{t}}_1, \ldots, \tilde{\mathbf{t}}_m]$ is defined recursively as $\tilde{\mathbf{t}}_1 \leftarrow \mathbf{t}_1$ and $\tilde{\mathbf{t}}_i \leftarrow \pi_i(\mathbf{t}_i)$ for $i \in [2, m]_\mathbb{Z}$, where π_i is the projection from $\mathbb{Z}^m$ onto $span(\tilde{\mathbf{t}}_1, \ldots, \tilde{\mathbf{t}}_{i-1})^\perp$. In most applications, the length $\left\|\tilde{\mathbf{T}}\right\|$ is more important than $\|\mathbf{T}\|$. See, e.g., [Bab86, Kle00, GPV08, CHKP10, Pei10].

The following proposition summarizes the second construction in [AP09] in its improved form [AP08].

Proposition 2.4 (Trapdoor Generation). *Let $C > 0$ and $\delta > 0$ be constants and let $q \geq 3$ be odd. Let $m_1 \geq d = (1+\delta)n\log(q)$, $m_2 \geq (4+2\delta)n\log(q)$, and $m \leftarrow m_1+m_2$. There is a PPT algorithm that outputs $(\mathbf{A}, \mathbf{T}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}^{m\times m}$ with the following properties.*

- $\mathbf{A}$ *is distributed within $m_2 q^{-\delta n/2}$ distance from uniform over $\mathbb{Z}_q^{n\times m}$;*
- $\mathbf{T}$ *is a basis of $\Lambda_q^\perp(\mathbf{A})$;*
- $\|\mathbf{T}\| \leq L = Cn\log(q) = \mathcal{O}(n\log(q))$ *with overwhelming probability;*
- $\left\|\tilde{\mathbf{T}}\right\| \leq \tilde{L} = 1 + C\sqrt{d} = \mathcal{O}(\sqrt{n\log(q)})$ *with overwhelming probability.*

2.7.3 Preimage-sampleable Trapdoor Functions

Recall the definition of trapdoor one-way functions TFF = (Kg, Eval, Inv) as defined in Section 2.3.2. Now, consider the algorithm $\mathsf{Eval}(pk, \cdot)$ to be non-injective in the second argument for all public keys pk. Then, Inv does not invert Eval, but rather samples from the set of valid preimages.

In this setting, we need to modify the security notion in $\mathsf{Exp}^{\mathsf{TRAP\text{-}OW}}_{\mathcal{A},\mathrm{TFF}}$ as follows. When $\mathcal{A}$ is challenged with $y \leftarrow \mathsf{Eval}(pk, x)$, it is supposed to output x in $\mathsf{Exp}^{\mathsf{TRAP\text{-}OW}}_{\mathcal{A},\mathrm{TFF}}$. Here, we merely ask $\mathcal{A}$ to output any x' from the correct domain, satisfying $y = \mathsf{Eval}(pk, x')$.

Gentry, Peikert, and Vaikuntanathan construct such a family PSTF = (Kg, Eval, SampleDom, SamplePre) of *preimage-sampleable trapdoor functions* in [GPV08]. There is also an ideal-lattice interpretation due to Stehlé, Steinfeld, Tanaka, and Xagawa [SSTX09], which we will use in Chapter 8. We defer the details.

Parameters: Let q, m, $\tilde{L}$ be as per Proposition 2.4 and let $\eta = \eta(n) = \theta(n)\tilde{L}$ for $\theta(n) = \omega(\sqrt{\log(n)})$ be the parameter of the distribution on the domain.[1]

Key Generation [Proposition 2.4]: $\mathsf{Kg}(1^n)$ outputs a public key $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a secret trapdoor matrix $\mathbf{T} \in \mathbb{Z}^{m \times m}$ that is a basis of $\Lambda_q^{\perp}(\mathbf{A})$ with $\left\|\tilde{\mathbf{T}}\right\| \leq \tilde{L}$.

Evaluation: $\mathsf{Eval}(\mathbf{A}, \mathbf{x})$, on input $\mathbf{x} \in \mathbb{Z}^m$, returns $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$.

Domain Sampling: $\mathsf{SampleDom}(m, \eta)$ returns $\mathbf{x} \in \mathbb{Z}^m$ with $0 < \|\mathbf{x}\|_\infty \leq \eta\theta(m)$ and $0 < \|\mathbf{x}\|_2 \leq \eta\sqrt{m}$; both with overwhelming probability.

Preimage Sampling: $\mathsf{SamplePre}(\mathbf{T}, \eta, \mathbf{y})$ outputs $\mathbf{x} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$ subject to $\mathbf{A}\mathbf{x} \equiv \mathbf{y}$ and $\|\mathbf{x}\|_2 \leq \eta\sqrt{m}$ (alternatively, $\|\mathbf{x}\|_\infty \leq \eta\theta(m)$) with overwhelming probability.

The sampling algorithms SampleDom and SamplePre are related to a modified version of Babai's nearest plane algorithm [Bab86] due to Klein [Kle00]. The sampling procedure is one of the bottlenecks in lattice-based schemes that require a trapdoor basis. Thus, efficiency improvements, such as Peikert's pre-computation approach in [Pei10], are very important.

The functions in PSTF compress the domain $D = \{\mathbf{x} \in \mathbb{Z}^m : \|\mathbf{x}\|_2 \leq \eta\sqrt{m}\}$ (or, $\{\mathbf{x} \in \mathbb{Z}^m : \|\mathbf{x}\|_\infty \leq \eta\theta(m)\}$), making them lose information about the input.

[1] The relevant distribution is a discrete Gaussian distribution on $\mathbb{Z}^m$ that can be sampled efficiently. The parameter η is akin to the standard deviation. For details, refer to [GPV08] as they are inconsequential for our work.

In particular, the preimages $\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{T}, \eta, \mathbf{y})$ of every $\mathbf{y} \leftarrow_{\$} \mathbb{Z}_q^n$ carry a high conditional minimum entropy $\omega(\log(n))$.

Interestingly, they are also one-way, collision-resistant, and claw-free under the assumption that SIS is hard for norm bounds $\nu \leq \mathrm{poly}(n)$ [GPV08]. In addition, for $(\mathbf{A}, \mathbf{T}) \leftarrow \mathsf{Kg}(1^n)$, the matrix $\mathbf{A}$ is within negligible statistical distance from uniform and the following two processes yield statistically indistinguishable random variables X and Y, i.e, $\Delta(X, Y)$ is negligible.

Forward: Sample $\mathbf{x} \leftarrow \mathsf{SampleDom}(m, \eta)$ and set $\mathbf{y} \leftarrow \mathsf{Eval}(\mathbf{A}, \mathbf{x})$. The output is $X \leftarrow (\mathbf{x}, \mathbf{y})$.

Backward: Pick $\mathbf{y} \leftarrow_{\$} \mathbb{Z}_q^n$ and sample $\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{T}, \eta, \mathbf{y})$. The output is $Y \leftarrow (\mathbf{x}, \mathbf{y})$.

In particular, the output of $\mathsf{Eval}(\mathbf{A}, \mathsf{SampleDom}(m, \eta))$ is distributed statistically close to the uniform distribution over $\mathbb{Z}_q^n$.

Chameleon Hash Functions Observe the similarity with chameleon hash functions from Section 2.3.3. There, it was also crucial that *forward* and *backward* sampling yield indistinguishable distributions. As already mentioned in [CHKP10], PSTF gives rise to a family of chameleon hash functions $\mathcal{C} = (\mathsf{Kg}, \Psi)$. We make the idea explicit.

Key Generation: $\mathsf{Kg}(1^n)$ selects $\mathbf{B} \leftarrow_{\$} \mathbb{Z}_q^{n \times m}$ and computes $(\mathbf{A}, \mathbf{T}) \leftarrow \mathrm{PSTF}.\mathsf{Kg}(1^n)$. It outputs the public description $(\mathbf{A}, \mathbf{B})$ for C and the trapdoor $\mathbf{T}$ for C^{-1}.

Randomness: Let $\Psi = \Psi_\eta$ be the output distribution of $\mathsf{SampleDom}$. The randomness ρ is generated via $\mathsf{SampleDom}(m, \eta)$.

Evaluation: $\mathsf{C}(msg, \rho)$, on input $msg \in \{0,1\}^m$ and $\rho \in \mathbb{Z}^m$, outputs $\mu \leftarrow \mathbf{A}\rho + \mathbf{B}msg \bmod q$. If $\|\rho\|_2 > \eta\sqrt{m}$, the output is $\perp$.

Inversion: $\mathsf{C}^{-1}(\mu, msg)$, on $\mu \in \mathbb{Z}_q^n$ and $msg \in \{0,1\}^m$, samples and outputs $\rho \leftarrow \mathsf{SamplePre}(\mathbf{T}, \eta, \mu - \mathbf{B}msg)$.

All requirements for chameleon hash functions are satisfied and a collision $(msg, \rho) \neq (msg', \rho')$ under C, with $\mathsf{C}(msg, \rho) \neq \perp$, yields a short non-zero lattice vector $\mathbf{v} \leftarrow (\rho \| msg) - (\rho' \| msg')$ in $\Lambda_q^{\perp}(\mathbf{A} \| \mathbf{B})$ because $(\mathbf{A} \| \mathbf{B})\mathbf{v} \equiv \mathbf{0}$ and $\|\mathbf{v}\|_2 \leq 2\eta\sqrt{m} + 2\sqrt{m}$.

Chapter 3

The Hardness of SIS and LWE in Practice

Essentially all modern lattice-based cryptography is based on the hardness of two average-case problems: SIS and LWE. These problems are conjectured to withstand quantum-computer and sub-exponential attacks. Furthermore, via worst-case to average-case reductions, they allow us to base security on the worst-case hardness of computational problems.

However, the above advantages come at a price. Usually, the bit lengths of the involved keys are $\Omega(n^2 \log^2(n))$. Fortunately, we can use ideal lattices, introduced by Micciancio [Mic07] as well as by Peikert and Rosen [PR06], that reduce the key size to $\mathcal{O}(n \log^2(n))$ bits. Still, in practice, choosing n as small as possible is crucial. To the best of our knowledge, there is no work that systematically deals with selecting secure parameters or analyzing the hardness of the employed assumptions. Indeed, the task is more involved than in the case of, say, RSA. Lattice cryptosystems have numerous parameters that affect security and dealing with n alone is not sufficient.

So far, only Micciancio and Regev [MR08], Lyubashevsky [Lyu09], as well as Lyubashevsky and Micciancio [LM08] have discussed practical parameters for their schemes. In [MR08, Lyu09], this choice is based on an interesting observation by Gama and Nguyen [GN08b]. They consider the Hermite Short Vector Problem HSVP with parameter $\delta > 0$ in lattices Λ of dimension d. There, the task is to find a vector $\mathbf{v}$ with $0 < \|\mathbf{v}\|_2 \leq \delta^d \det(\Lambda)^{1/d}$. In [GN08b], the authors analyze "random lattices" according to the Goldstein-Mayer distribution [GM03] that are considered to provide hard instances of HSVP. Their observation is that δ is the dominating parameter and that d only plays a minor role. They conjecture that HSVP seems feasible for $\delta \approx 1.01$ and "totally out of reach" for $\delta < 1.005$ in dimensions $d \geq 500$ if the lattice does not have a special structure. We build upon their work and refine their analysis using precisely the distribution of lattices that is relevant in cryptography.

The good news is the "hardness estimate" δ can be determined from the security proof for the cryptosystem. The bad news is that *cryptographic* q-ary lattices have a particular structure that can be exploited in attacks. Micciancio and Regev describe such an attack in [MR08]. The bottom line is that solving δ-HSVP in q-ary lattices of dimension m is only as hard as solving δ'-HSVP in dimension $d < m$ and $\delta' > \delta$. Thus, HSVP becomes strictly easier in q-ary lattices because there is a certain "slack" in the required attack dimension.

With this knowledge, two unsatisfying options remain. The *first* involves the worst-case to average-case reduction. One could interpret the results of Gama and Nguyen as observations about worst-case SIVP in dimension n, while the best attack against the cryptosystem needs to work in dimension $\Omega(\sqrt{n \log(n)})$ [MR08]. Hence, this approach would work but it is overly conservative and the resulting parameters

would be impractical.

The *second* option is using the results of Gama and Nguyen in dimension d, while demanding that $\delta < 1.01$ for security against current means. Basically, this is the methodology in [MR08, Lyu09] but it only offers a *yes*/*no* certificate, i.e., a given parameter set is either secure or insecure. In particular, it does not offer security levels, such as 100 bits, meaning that the attack effort should be close to 2^{100} storage times computation units.

With our work, we intend to provide a *third* option, with a focus on lattice-based encryption [Reg09, GPV08, Pei09, SSTX09, LPR10] and signature schemes [GPV08, SSTX09, Lyu09, LM08, CHKP10, Boy10] because they are the main building blocks of public-key cryptography. Nevertheless, our results can be easily applied to more advanced schemes, such as identity-based encryption [GPV08], oblivious transfer [PW08, PVW08], collision resistant hashing [LM06], secret key delegation [CHKP10], and others. This versatility is also demonstrated in Chapter 4 and Chapter 6.

We do not consider ad-hoc constructions, such as NTRU [HPS98], that fall outside the category of schemes motivated by Ajtai's work. The lattices that correspond to attacks on NTRU have a particular structure and contain *essentially* one unusually short "trapdoor" vector. Random q-ary lattices do not admit such a structure.

Apart from choosing secure parameters, we often wish to compare schemes with regard to their security level. Say, we have scheme X and a new scheme Y, which is more efficient than X in the sense that its public key is smaller, but at the expense of a stronger assumption. For a fair comparison, we first need a methodology to generate parameter sets that yield comparable security levels. Now, two things could happen: (1) the improvements in Y are still noticeable or (2) due to the stronger assumption, Y requires, say, a larger dimension that effectively nullifies the proposed improvement.

Our Contribution Inspired by the works of Lenstra and Verheul [LV01] and the subsequent update by Lenstra [Len05], we propose a unified methodology for estimating security and selecting secure parameters for *all* modern lattice-based cryptography. To this end, we adopt the notion of dollar-days, i.e., equipment cost in dollar *times* attack time in days, as introduced in [Len05]. Our methodology in Section 3.1 also includes different attacker types, ranging from a resource-constrained "Hacker" to an all-powerful "Intelligence agency".

We follow a modular three-tier approach: core analysis, experiments, and application.

Tier 1: At the core, there are our conjectures and observations about how the various parameters for LWE and SIS influence the hardness of these problems in Section 3.2. In addition, via the duality of LWE and SIS, we translate LWE instances into the language of SIS. For both, we manage to distill the hardness into one single parameter.
Tier 2: Then, we establish a relation between the attack effort in practice and this single hardness parameter by running a large number of experiments. In particular, this relation offers a way to determine the equivalent symmetric bit-security. This is done by running practical attacks on feasible instances of SIS, followed by a conservative extrapolation in Section 3.2. Like Gama and Nguyen [GN08b] did in a different context, we observe that the complexity of lattice-based attacks is mainly governed by δ. Therefore, we propose a function $T(\delta)$ that estimates the attack complexity in dollar-days for $\delta \in (1, 1.02]$ in Section 3.2. There, we also demonstrate that current records in practical lattice basis reduction support our findings. The underlying experiments can be easily replaced as soon as there are more powerful algorithms. The other two tiers stay unchanged. Notice that new experiments are *not* required if the algorithmic improvements are already covered by our double-Moore Law, i.e., we already anticipate new attacks and more powerful computing devices. Interestingly, our estimation shows that, today, $\delta = 1.009$ is potentially reachable with an effort of 40 million dollar-days. However, even a powerful intelligence agency with over 100 billion dollar-days of resources should not be able to reach $\delta = 1.005$ before the year 2050. We draw the following main conjecture.

Conjecture (Main conjecture)**.** *Let $n \geq 128$, $q = q(n)$, $m = m(n)$, and $\nu = \nu(n)$. Let $T(\delta) = 10^{-15}2^{1/(\log_2(\delta)^{1.001})}$ dollar-days be a cost function for $\delta > 1$.*

Solving SIS(n, m, q, ν) involves a q-ary lattice of dimension $d = \min\{x \in \mathbb{N} : q^{2n/x} \leq \nu\}$ and an attack effort of at least $T(\delta)$ with $\delta = \sqrt[d]{\nu/q^{n/d}}$.

As for LWE, let $\Psi = \Psi_\alpha^m$ be a component-wise m-dimensional Gaussian distribution with standard deviation $q\alpha/\sqrt{2\pi}$. Let $\nu^* = 1.5\sqrt{2\pi}/\alpha$. Then, solving LWE$(n, m, q, \Psi)$ involves a q-ary lattice of dimension $d = \min\{x \in \mathbb{N} : q^{2n/x} \leq \nu^*\}$ and an attack effort of at least $T(\delta)$ with $\delta = \sqrt[d]{\nu^*/q^{n/d}}$.

Tier 3: The third part is the application of our framework to cryptographic schemes in Section 3.3. We evaluate the security of proposed parameter sets in the literature and find that most do not provide sufficient security. Similarly, we use our formulas in the reverse direction to output parameter sets for a given security level.

Thus, we can make absolute statements about individual cryptosystems, saying that schemes X with parameter set $P(X)$ is secure against a certain type of attacker $\mathcal{A}$ until the year 2030. In addition, since we propose a unified framework for both

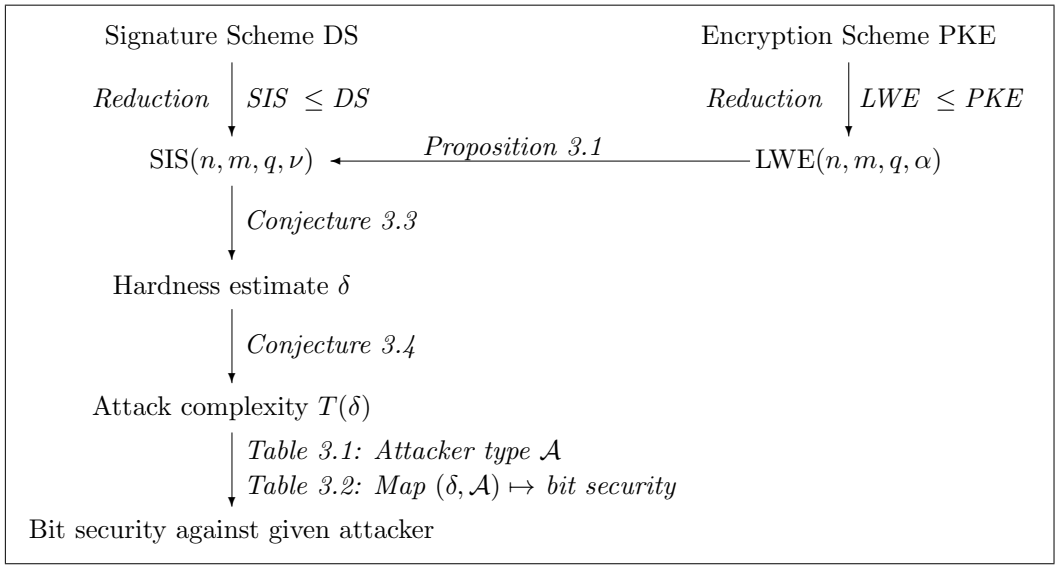


Figure 3.1: User's guide to estimating the bit security of lattice-based signature and encryption schemes.

SIS and LWE, we can also make relative statements across different SIS- and LWE-based schemes. For example, saying that SIS scheme X with parameters $P(X)$ is *more*, *less*, or *as* as secure as LWE scheme Y with parameters $P(Y)$. Figure 3.1 summarizes our analysis and provides a user's guide to estimating the security of a given parameter set.

This chapter is a reprint of the essential parts of [RS10a]. The dissertation author was the primary investigator and author of this paper. The full report is constantly updated and covers essentially all published signature and encryption schemes.

3.1 Methodology

In order to grasp lattice reduction algorithmically, the notion of Hermite-SVP (HSVP) approximation seems more adequate than that of approximate SVP. A simple reason, in practice, is that we do not know the first successive minimum λ_1 when confronted with an arbitrary lattice basis. Therefore, it is impossible to verify the SVP-condition $\|\mathbf{v}\|_2 \leq \gamma\lambda_1(\Lambda)$ directly. In contrast, HSVP asks for a non-zero vector that satisfies $\|\mathbf{v}\|_2 \leq \delta^{\dim(\Lambda)} \det(\Lambda)^{1/\dim(\Lambda)}$ for a given $\delta > 0$, which can be easily

verified because all quantities are known. In fact, the condition is further simplified in q-ary lattices and it becomes: find $\mathbf{v} \in \Lambda_q^\perp(\mathbf{A})$ such that $0 < \|\mathbf{v}\|_2 \leq \delta^m q^{n/m}$.

Concerning the hardness of this problem, the lattice dimension certainly plays a role but Nguyen and Gama demonstrate in [GN08b] that δ is the dominating parameter. Although their analysis is based on Goldstein-Mayer (GM) lattices [GM03], which fundamentally differ from q-ary lattices, this basic observation holds in q-ary lattices as well. Hence, we build upon their work and extend it with a fine-grained hardness estimate in the sense of having a cost function T.

To this end, we interpret instances of SIS and LWE as instances of HSVP. Then, we conduct a series of experiments with lattice basis reduction algorithms to determine how the various parameters in SIS and LWE influence their hardness. Based on these experiments, we extrapolate the attack complexity. The extrapolation takes future algorithmic and technological progress into account and we arrive at an arguably conservative cost function T.

From SIS to HSVP Each instance of SIS can be interpreted as an instance of the Hermite-SVP. Given an instance $\mathbf{A}$ of $\mathrm{SIS}(n, m, q, \nu)$, we compute $\delta = \sqrt[m]{\nu/q^{n/m}}$ and ask the Hermite-SVP solver to find $\mathbf{v}$ with $0 < \|\mathbf{v}\|_2 \leq \delta^m q^{n/m}$. However, this direct translation is not the best possible attack. In [MR08], Micciancio and Regev point out that one can solve the same problem in a significantly lower lattice dimension. They assume the existence of a δ-HSVP solver for a fixed δ. Then, they argue that the optimum dimension for solving SIS with (n, m, q) with this solver is $d = \min\{\sqrt{n \log(q)/\log(\delta)}, m\}$. Now, one removes $m - d$ random columns from $\mathbf{A}$ to obtain $\mathbf{A}'$, reduces the d-dimensional lattice bases of $\Lambda_q^\perp(\mathbf{A}')$, and pads a short vector therein with zeros. The result is a rather sparse vector of norm $\leq \delta^d q^{n/d}$ in $\Lambda_q^\perp(\mathbf{A})$.

Unfortunately, this approach does not model the most likely attacker, which will take ν into account and employ stronger and stronger HSVP solvers until a sufficiently short vector is found. Therefore, we need a re-interpretation of the approach taken in [MR08] that involves ν instead of a fixed "capability" δ. This re-interpretation allows us to *normalize* $\mathrm{SIS}(n, m, q, \nu)$ by removing the slack in the dimension parameter m. The resulting distribution of lattices is what we will analyze by directly applying lattice basis reduction. We defer the details to Section 3.2.

From LWE to HSVP While solving the search LWE problem also immediately solves the corresponding decision problem, the reverse direction only holds via a polynomial-time reduction that is not "sample-preserving". Thus, we choose to

attack the decision problem as it presents the easier problem and yields a lower bound for search LWE.

The most natural approach to distinguish an LWE sample $(\mathbf{A}, \mathbf{b})$ from uniform seems to be solving an instance of the SIS problem. Evidence for this connection can be found in [MR08] and [SSTX09]. Let $\Psi = \Psi_\alpha^m$ be the standard noise distribution for LWE as per [Reg09], i.e., a component-wise discretized Gaussian distribution with standard deviation $q\alpha/\sqrt{2\pi}$. Then, [MR08] essentially yields the following proposition.

Proposition 3.1 (From LWE to SIS)**.** *LWE(n, m, q, α), interpreted as an instance of SIS(n, m, q, ν) requires an SIS-solver for $\nu \ll 1.5\sqrt{2\pi}/\alpha$ to be successful.*

Given an LWE instance $(\mathbf{A}, \mathbf{b})$ in the lattice $\Lambda_q(\mathbf{A})$, the idea is to find a short vector $\mathbf{v}/q$ in its dual lattice $1/q\Lambda_q^\perp(\mathbf{A})$. By the definition of duality, $\langle \mathbf{b}, \mathbf{v}/q \rangle$ is an integer if $\mathbf{b}$ is in $\Lambda_q(\mathbf{A})$. Similarly, if it is close to $\Lambda_q(\mathbf{A})$, $\langle \mathbf{b}, \mathbf{v} \rangle$ will be close to an integer. Indeed, if $\mathbf{b}$ is an LWE sample, we have $\langle \mathbf{b}/q, \mathbf{v} \rangle \equiv \langle \mathbf{A}^t\mathbf{s}/q + \mathbf{e}/q, \mathbf{v} \rangle \equiv \langle \mathbf{A}^t\mathbf{s}/q, \mathbf{v} \rangle + \langle \mathbf{e}/q, \mathbf{v} \rangle \equiv \langle \mathbf{e}/q, \mathbf{v} \rangle \pmod 1$ because $\mathbf{Av} \equiv \mathbf{0} \pmod q$.

To verify that $\mathbf{e}$ is short, we require $\mathbf{v}$ to be short. If it is at least $1.5\sqrt{2\pi}/\alpha$, the above residual scalar product is distributed like a normal distribution with mean zero and standard deviation 1.5. The observed fractional part thereof is very close to uniform.

As a conservative measure, we will assume in the following that an SIS-solver with $\nu = 1.5\sqrt{2\pi}/\alpha$ is already a successful distinguisher for LWE. This makes it even more conservative than using the approach recently taken in [LP10].

Lenstra's Heuristic The authors of [ECR10] describe an attacker model with attacker classes according to [BDR+96]; a subset of these classes is shown in Table 3.1. We add an attacker called "Lenstra", with an amount of 40M dollar-days, which was the value for a suitable attacker proposed by Lenstra in [Len05]. Let n be the security parameter and assume the best attack against a given cryptosystem takes $t(n)$ seconds on a machine that costs d dollars. Then, the total "cost" of the attack is $T(n) = d \cdot t(n)/(3600 \cdot 24)$ dollar-days (DD). This notion is particularly interesting when estimating attack cost against lattice cryptography, where attacks may be parallelized with a time-money tradeoff.

Assuming we have an estimate for the function $T(n)$ for attacks against lattice-based cryptosystems. Then, we can find the optimum n^* such that $T(n^*) \geq T_{2009}$, where T_{2009} is chosen according to the last column of Table 3.1. We choose 2009 as a reference date here because the employed hardware was bought in that year.

Attacker class	Budget	Time	Dollar-days
Hacker	\$400	1 d	400 DD
Lenstra			40M DD
Intelligence agency	\$300M	360 d	108B DD

Notation: "d" stands for days, "M" for million, "B" for billion, and "DD" for dollar-days. For "Lenstra", there is not budget-time separation in [Len05].

Table 3.1: Attacker classes and corresponding budget for each attacker.

Estimating Future Developments First of all, we consider Moore's Law, which states that computing power doubles every 18 months. Secondly, we want to take cryptanalytic developments into account. Thus, we apply a combined degradation function $2^{-12/9}$ that Lenstra calls "double Moore Law". This is motivated by the observed algorithmic progress in the area of integer factorization. As for lattice basis reduction, the algorithmic progress for practical strong algorithms, such as BKZ, is hard to judge. While, there are recent results [GHGKN06, GN08a, GNR10] showing that progress is indeed possible, there are no public implementations that beat BKZ in practice.

The above condition $T(n^*) \geq T_{2009}$ only yields secure parameters for the year 2009. For the year y, n^* needs to satisfy the inequality $T(n^*) \geq T_{2009} \cdot 2^{(y-2009)\cdot 12/9}$.

Asymmetric primitives are often combined with symmetric ones. Hash functions are necessary to sign long documents and block ciphers allow efficient hybrid encryption. We assume that these primitives are available at any given time in the future and that they are only affected by Moore's Law. Unlike public-key primitives, block ciphers and hash functions can easily be replaced if there is a new attack.

3.2 Analysis

Given that we can phrase the LWE problem in the language of SIS, we restrict our analysis to the SIS problem. An attacks against SIS with a matrix $\mathbf{A}$ involves a q-ary lattice $\Lambda = \Lambda_q^{\perp}(\mathbf{A})$ of dimension $m = \Omega(n \log(n))$ and a scheme-specific norm bound ν, which can be obtained by studying the security proof.

The main goal of this section is to determine the effort T (in dollar-days) that is required today for mounting such an attack. From there, we can apply Lenstra's Heuristic to predict future developments. In Section 3.2.3, we show how to use

our findings in combination with symmetric primitives. There, we also relate the hardness of SIS with symmetric bit security. We conclude this section by showing that current records in lattice basis reduction support our analysis (cf. Section 3.2.4).

We have conducted experiments on up to 100 random q-ary lattices per dimension $m \in \{100, 125, 150, 175, 200, 225, 250, 275, 300\}$ and prime q with $n^c \leq q \leq 2n^c$ for $c \in \{2, 3, 4, 5, 6, 7, 8\}$. These parameters alone also determine n if we demand that $m > n \log_2(q)$. This setting covers even the hardest instances of SIS, where we demand $\nu = \sqrt{m}$. The existence of such vectors can be verified with a pigeonhole argument because the function $f_{\mathbf{A}}(\mathbf{v}) = \mathbf{A}\mathbf{v} \bmod q$ admits collisions $(\mathbf{v}, \mathbf{v}') \in (\{0,1\}^m)^2$ if $q^n/2^m < 1$. Such a collision yields $\mathbf{v} - \mathbf{v}' \in \Lambda_q^\perp(\mathbf{A})$ with $\|\mathbf{v} - \mathbf{v}'\|_2 \leq \sqrt{m}$.

As mentioned earlier, we do not attack the "full" lattice, but rather work in a sub-dimension d. Instead of assuming a fixed-capability attacker that solves δ-HSVP in dimension $d = \sqrt{n \log(q)/\log(\delta)}$, we propose the following approach and let d be determined only by n, q, and ν.

Proposition 3.2 (Normalization of q-ary Lattices)**.** *Let $n \geq 128$, $q \geq n^2$, and $\nu < q$. Let S be a δ-HSVP solver for variable $\delta > 1$. The optimal dimension for solving SIS(n, m, q, ν) with S is $d = \min\{x \in \mathbb{N} : q^{2n/x} \leq \nu\}$.*

The restrictions $n \geq 128$ and $\nu < q$ rule out easy cases and we demand $q \geq n^2$ because the worst-case to average-case reduction typically demands $q = \omega(n)$.

Proof. Notice that when removing $m - d$ random columns from $\mathbf{A}$ to form a matrix $\mathbf{A}' \in \mathbb{Z}_q^{n \times d}$, the resulting q-ary lattice $\Lambda_q^\perp(\mathbf{A}')$ still has determinant q^n because $\mathbf{A}'$ generates $\mathbb{Z}_q^n$ with high probability. Observe that $d > 2n$ as otherwise $q^{2n/d} \geq q > \nu$, which means that we cannot expect to find useful (short enough) vectors.

The solver S finds lattice vectors of norm at most $2^{2\sqrt{n \log(q) \log(\delta)}}$ (cf. [MR08]), which must not exceed ν. Hence, $\log(\delta) \leq \log^2(\nu)/(4n \log(q))$. The solver S is most efficient for the largest permissible δ. Then, the optimal dimension for running the solver is $d = \sqrt{n \log(q)/\log(\delta)} = 2n \log(q)/\log(\delta)$, which is the minimum d such that $q^{2n/d} \leq \nu$. □

Note that, as mentioned in the above, the minimum attack dimension is $d > 2n \geq 256$. Hence, special algorithms that efficiently reach smaller δ in "easy" dimensions < 256, do not contradict our analysis. To sum up, our analysis is based on the following conjecture.

Conjecture 3.3. *For every $n \geq 128$, prime $q \geq n^2$, $m = \text{poly}(n)$, and $\nu < q$, the best known approach to solve SIS with parameters (n, q, m, ν) involves solving δ-HSVP in dimension $d = \min\{x \in \mathbb{N} : q^{2n/x} \leq \nu\}$ with $\delta = \sqrt[d]{\nu/q^{n/d}}$.*

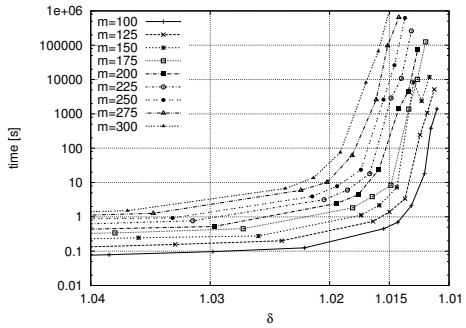
3.2.1 Experimental Data

In our experiments, we have analyzed the running time of BKZ [SE94] with double floating-point precision, the best publicly available scalable HSVP-solver, as implemented in Shoup's NTL [Sho] on a $1,000 machine (AMD Opteron CPU, running at 2.4 GHz, bought in 2009). We apply BKZ in the sub-dimension d with an increasing block-size parameter, i.e., with decreasing δ, until a vector of the desired length is found. Note that our experiments only involve block-size parameters ≤ 30 in order to avoid a known erratic behavior of BKZ in practice. If an attack was not possible with an admissible block size, we have not taken the data sample into account. Also, performing the experiments in rather small dimensions, we give quite a conservative hardness estimate. Our first observation is that q plays a minor role. To see this, compare figures 3.2(a) ($q \approx n^2$) and 3.2(c) ($q \approx n^8$). The graphs show the same shape. This also holds for $n^2 \leq q \leq n^8$. However, increasing q makes the problem slightly harder. The impact of the dimension is noticeable because n increases with it. As a result, we see shifted copies of essentially the same graph. The interesting part of the figures is where δ is smaller than 1.015, i.e., the right-hand side of the graphs. Here, the impact of the parameter δ is compelling, and much more noticeable than the impact of the other parameters. Thus, we can consider δ to be the dominating and main security parameter.

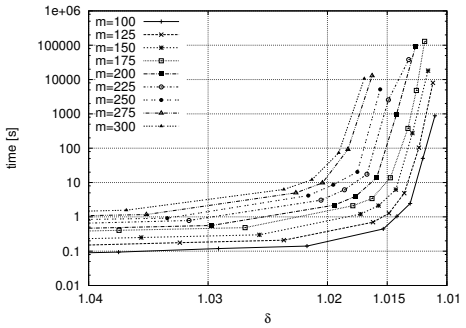
We have chosen $m = 175$ and $q \approx n^3$ (cf. Figure 3.2(b)) as our reference for extrapolation. In order to compensate for these rather small dimensions, we have chosen $q \approx n^3$ and not n^2. Still, based on our experiments in small dimensions, we arrive at fairly conservative estimates. The extrapolation in Figure 3.2(d) was used to determine the hardness of attacks against SIS. For the interesting area where $\delta < 1.015$, the "extrapolated attack complexity" function $T(\delta)$ nicely approximates the data samples.

More precisely, the cost function in dollar-days is of the form $T(\delta) = a2^{1/(\log_2(\delta)^b)}$, for real constants a, b. Using a least-squares approximation, we draw the following conjecture.

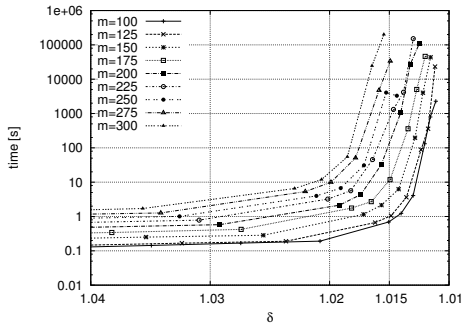
Conjecture 3.4. *Let all other parameters and relations as in Conjecture 3.3. For any $\delta \in (1, 1.015]$, solving δ-HSVP (in normalized q-ary lattices) involves an effort of at least $T(\delta) = 10^{-15}2^{1/(\log_2(\delta)^{1.001})}$ dollar-days.*



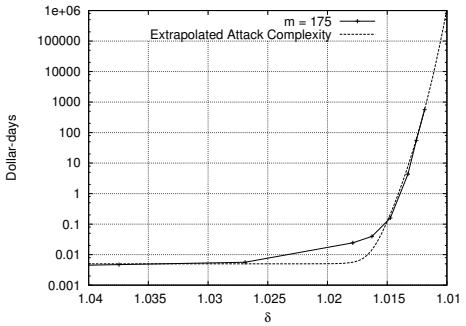
(a) Logarithmic running time in seconds for prime $q \approx n^2$ and selected $100 \leq m \leq 300$ and $1.01 < \delta \leq 1.04$.



(b) Logarithmic running time in seconds for prime $q \approx n^3$ and selected $100 \leq m \leq 300$ and $1.01 < \delta \leq 1.04$.



(c) Logarithmic running time in seconds for prime $q \approx n^8$ and selected $100 \leq m \leq 300$ and $1.01 < \delta \leq 1.04$.



(d) Logarithmic effort in dollar-days (data & extrapolation) for prime $q \approx n^3$, $m = 175$, and $1.01 < \delta \leq 1.04$.

Figure 3.2: Logarithmic time complexity for solving δ-HSVP in different dimensions and for different moduli q. The x-axis corresponds to the hardness estimate δ.

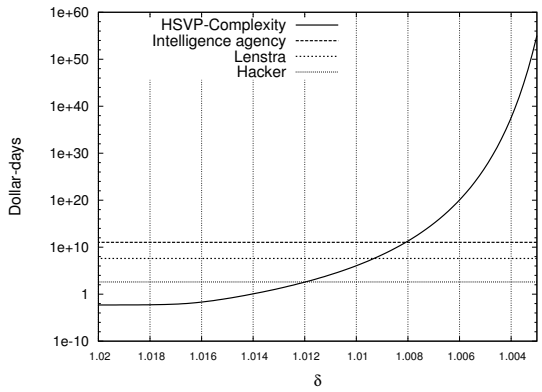


Figure 3.3: Estimated time complexity of δ-HSVP for $\delta \in [1.003, 1.02]$. The plots include horizontal lines, illustrating today's power of different attacker types.

Extrapolating T for smaller δ yields Figure 3.3. The horizontal bars correspond to today's capabilities of the attacker types in Table 3.1.

3.2.2 Applying Lenstra's Heuristic

Fix an attacker type $\mathcal{A}$ and let $\delta_{\mathcal{A}}$ be infeasible for $\mathcal{A}$ today. Assuming the Lentra Heuristic in conjunction with the "double Moore Law", we demand $T(\delta) \geq T_{2009} \cdot 2^{12(y-2009)/9}$ for $T_{2009} = T(\delta_{\mathcal{A}})$ for δ to be secure until the year y. Obviously, the inequality can also be used to compute a year y until which a given parameter δ can be considered secure against a given attacker.

Note that the inverse function is $T^{-1}(t) = 2^{(1/(\log_2(t)\cdot 10^{15}))^{1/1.001}}$, where t is the amount of dollar days available. For example, let $\mathcal{A}$ = "Int. agency". Compared with the year 2009, it can afford $t = 108 \cdot 2^{124/3}$ billion dollar-days in 2040. Thus, we require $\delta \leq T^{-1}(t) = 1.00548$ for infeasibility until the end of 2040. Vice versa, if an attack requires $\delta \leq 1.00548$, the corresponding lattice problem is at least intractable until the end of 2040. Table 3.2 provides an overview of hard values for δ for the different attacker types until 2100. This table also allows a mapping between symmetric security and security parameters for lattice cryptography. In addition, we include a column "standard" for a standard hash function (SHA-1) and a standard block cipher (AES-128). The resulting parameter sets can be considered secure against *non-quantum* adversaries until 2018.

year	Standard (2018)	2010	2020	2030	2040	2050	2060	2070	2080	2090	2100
bit security	SHA/AES	75	82	88	95	102	108	115	122	128	135
$\boldsymbol{\lambda}$	160	225	246	264	285	306	324	345	366	384	405
$\boldsymbol{\kappa}$	128	150	164	176	190	204	216	230	244	256	270
Hacker	1.00993	1.01177	1.00965	1.00808	1.00702	1.00621	1.00552	1.00501	1.00458	1.00419	1.00389
Lenstra	1.00803	1.00919	1.00785	1.00678	1.00602	1.00541	1.00488	1.00447	1.00413	1.00381	1.00356
Int. agency	1.00710	1.00799	1.00695	1.00610	1.00548	1.00497	1.00452	1.00417	1.00387	1.00359	1.00336

The upper rows present recommended post-quantum secure symmetric key size κ and hash function length λ. Each of the lower cells contains an upper bound for the HSVP-parameter δ, such that this problem is computationally hard for the given attacker (row) until the end of a given year (column). *According to Proposition 3.2 solving δ-HSVP needs to be infeasible in dimensions $d \geq 256$.*

Table 3.2: Infeasible parameters δ for HSVP.

3.2.3 Post-quantum Secure Hash Functions and Symmetric Ciphers

Encryption schemes and hash functions are rarely used without block ciphers and collision-resistant hash functions, respectively. Since we want to propose parameters for the post-quantum era, we also want the symmetric ciphers and hash functions to be secure in this setting. In consequence, we need to take Grover's search algorithm for quantum computers into account [Gro96]. Basically, its effect is that we have to double the key length of block ciphers that would be classically secure. The output length of such hash functions has to be multiplied with 3/2. Note that is a pessimistic view on security as pointed out by Bernstein [Ber09].

As a simplification, we choose the symmetric parameters independently of the attacker type. A natural extension of our work would be to let λ and κ be functions of the attacker's resources. Here, we use the simple Moore Law and the assumption that DES was secure in the year 1982, even against the strongest attacker. Then, $\kappa \geq 2\lceil 56 + 12(y - 1982)/18 \rceil$ is the proposed symmetric key length and $\lambda \geq 3\kappa/2$ is the proposed output length for hash functions. Using these formulas, we obtain the recommendations in Table 3.2.

This concludes the analysis. Table 3.2 and Conjecture 3.4 provide all the necessary tools for estimating the security of all SIS and LWE-based cryptosystems. It also shows the equivalent level of symmetric security.

3.2.4 Comparison with Known Records in Lattice Reduction

There are two public challenges for hard lattice problems available online. The first — the "SVP challenge"[1] — is used to test exact, non-approximative SVP solvers. The second — the "Ajtai challenge"[2] — is used to evaluate the hardness of SIS. We briefly summarize the current state of both challenges at the time of writing this thesis.

SVP Challenge The challenge involves Goldstein-Mayer lattices in small dimensions of, say, < 200. The best contestants have found vectors of Euclidean length 2781 in dimension 112, which corresponds to $\delta = 1.009$. However, a success in this challenge does not have any immediate implication in our context, as it applies to a different type of lattice and the dimensions involved are small.

Ajtai Challenge The setup in this challenge is very similar to our setup here. It contains various simplifications, though. The details are described in [BLR08] and updated in [BLRS08]. The dissertation author was the principal investigator and author of these papers.

In the challenge itself, we use the simplified setup: $n \in \mathbb{N}$, $q = n$, and $m \approx n \log(n)$. The corresponding matrix $\mathbf{A}$ is generated from the digits of π to prevent it from containing a trapdoor. The initial task is to solve $\mathrm{SIS}(n, m, q, \nu)$ for $\nu = n$ in $\Lambda_q^\perp(\mathbf{A})$. Then, the participants are asked to find increasingly shorter solutions.

Notice that the instances in the challenge are easier than the instances used in this chapter. Hence, it yields lower bounds for the expected attack complexity $T(\delta)$.

The best participants have found vectors of length ≈ 107 in dimension $m = 750$. This corresponds to $\delta = 1.0103$ and an optimal attack dimension of $d = 229$ (cf. Proposition 3.2). Even though this result is still outside the relevant range for our analysis ($d \geq 256$), it confirms our estimates, saying that today, a "Hacker" should be able to solve the problem for $\delta \approx 1.011$ and the adversary "Lenstra" might even solve it for $\delta \approx 1.009$ in 2010 (cf. Table 3.2).

3.3 Applying the Framework

After having established a basic framework for estimating future developments in lattice basis reduction as well as the projected attack complexity against SIS and

[1] `http://www.latticechallenge.org/svp-challenge`
[2] `http://www.latticechallenge.org`

LWE, we will demonstrate how to use our framework as a tool.

There are essentially two "directions". In the "forward" direction, we can take a cryptographic parameter set and an attacker type as input and output an equivalent security level or even a prediction of how long this parameter set can be considered secure.

When working in the "reverse" direction, we analyze a given scheme's parameters and their relations as well as the corresponding worst-case to average-case reduction and, on input a year and an attacker type, output a set of feasible, concrete parameters that can be considered secure against the given attacker type until the given year.

As mentioned before, we can also make relative statements as follows: Given an SIS scheme X with parameters (n, q, m, ν) and an LWE scheme Y with parameters (n, q, m, α), we can compute their hardness parameters δ_X and δ_Y. If $\delta_X < (>)\delta_Y$, the instance of X is more (less) secure than the instance of Y.

3.3.1 Analysis of Proposed Parameter Sets

In this section, we will only apply our framework in the first sense, i.e., to analyze the (few) parameter sets that have been proposed in literature so far, regarding their exact security level. More concretely, we estimate the security of the parameters presented for LWE encryption in [MR08], to Lyubashevsky's Fiat-Shamir signature scheme in [Lyu09] (cf. Table 3.3), and to the one-time signature scheme due to Lyubashevsky and Micciancio [LM08]. Observe that neither of these authors make claims about the exact security of their proposals.

For SIS-based schemes [LM08, Lyu09], we analyze the corresponding security proofs to determine the relevant SIS parameters. For LWE [MR08], we compute the corresponding SIS parameters as outlined in Section 3.1.

Since the parameter sets given in [MR08] (see Table 3.3) were specifically chosen to be secure against attackers that can solve HSVP for $\delta \geq 1.01$, they do not provide sufficient security against the medium adversary "Lenstra" and even the "Hacker" should be able to break them by 2020.

For the Fiat-Shamir type signature scheme in [Lyu09], we compute the SIS norm parameter $\nu = 2nmd_s d_c\sqrt{mn}$ in the Euclidean norm, where d_s is the infinity norm of signature keys and d_c controls the hashed message length.

The parameters in [Lyu09] are based on an assumed hash length of 160 bit, therefore the underlying hash function would only be secure until year 2018 (without taking quantum adversaries into account). However, the lattice parameters are quite reasonable as shown in Table 3.3. All but the first parameter set provide some

n	136	166	192	214	233	233
q	2003	4093	8191	16381	32749	32749
α	0.0065	0.0024	0.0009959	0.00045	0.000217	0.000217
ν	5.8e2	1.6e3	3.8e3	8.4e3	1.7e4	1.7e4
d	326	376	421	460	497	497
δ	1.0098	1.0098	1.0099	1.0099	1.0099	1.0099
year	2006	2006	2005	2005	2005	2005
bit	72	72	72	72	72	72

n	512	512	512	1024
q	$2^{31.727}$	$2^{59.748}$	$2^{95.747}$	$2^{95.872}$
m	4	5	8	8
ν	5.6e8	1.3e10	2.6e10	6.4e10
d	1118	1823	2835	5471
δ	1.0091	1.0064	1.0042	1.0023
year	2010	2035	2077	2180
bit	75	92	120	188

d is the attack dimension and δ is the corresponding hardness estimate. "year" denotes the expiration year of the parameter set w.r.t. the attacker "Lenstra" and "bit" denotes the corresponding "bit security". ν is given in scientific notation, where "XeY" means $X \cdot 10^Y$.

Table 3.3: Security estimate for the parameters given in [MR08] (left) and [Lyu09] (right).

security margin and with our framework, we can actually estimate how large it is.

The authors of [LM08] propose an exemplary parameter set for their one-time signature scheme. They let $n = 512$, $q = 2^{27}$, $m = 9$, and argue that the parameter set is likely to be insecure. Now, we can analyze it explicitly as an SIS instance with $\nu = 20q^{1/m}n\log^2(n)\sqrt{mn} \approx 2.2 \cdot 10^8$ in the Euclidean norm. The attack dimension would be $d = 999$ and the hardness estimate is $\delta = 1.0097$. Hence, it would be insecure against the attacker "Lenstra" and the "Hacker" is expected to break it by the year 2020. On the positive side, our framework yields a similar parameter set with $n = 1024$ that is conjectured to be secure until 2050 against "Lenstra".

3.3.2 Secure Parameter Sets

To demonstrate the constructive power of our framework, we propose an exemplary parameter set for Regev's LWE encryption scheme [Reg09]. Two further applications are in Chapter 4, where we apply our framework to estimate secure parameter sets for our lattice-based blind signature scheme, and Chapter 6, where we estimate the net advantage of our aggregate signature scheme. Moreover, in the full report [RS10a], we propose parameter sets for essentially all published signature and encryption schemes.

Multi-bit LWE

The κ-bit version of Regev's LWE cryptosystem with standard noise distribution $\Psi = \Psi_\alpha^{m\times\kappa}$ works as follows.

Secret Key: $\mathbf{S} \leftarrow_\$ \mathbb{Z}_q^{n\times\kappa}$, i.e, $n\kappa\log_2(q)$ bits.

Public Key: $\mathbf{A} \leftarrow_\$ \mathbb{Z}_q^{n\times m}$, $\mathbf{P} \leftarrow \mathbf{A}^t\mathbf{S} + \mathbf{E} \in \mathbb{Z}_q^{m\times\kappa}$ for $\mathbf{E} \leftarrow_\Psi \mathbb{Z}_q^{m\times\kappa}$. The matrix $\mathbf{A}$ can be the same for all users, e.g., generated from the random bits of π. Using the HNF technique of [Mic01], the key is reduced to $(m-n)\kappa\log_2(q)$ bits.

Plaintext: $\mathbf{k} \in \mathbb{Z}_2^\kappa$.

Ciphertext: $\mathbf{u} \leftarrow \mathbf{A}\mathbf{a} \in \mathbb{Z}_q^n$, $\mathbf{c} \leftarrow \mathbf{P}^t\mathbf{a} + \mathbf{k}^{\frac{q-1}{2}}$, where $\mathbf{a} \leftarrow_\$ \{-\lfloor r/2\rfloor, \ldots, \lceil r/2\rceil\}^m$, $r \geq 1$. The ciphertext has $(n+\kappa)\log_2(q)$ bits.

Decryption: $\mathbf{c} - \mathbf{S}^t\mathbf{u} \approx \mathbf{k}^{\frac{q-1}{2}}$.

Ciphertexts are unconditionally uniform due to the leftover-hash lemma [HILL99] and the public key is computationally indistinguishable from uniform under the decision LWE assumption. We set $\alpha \leftarrow 1/(30\sqrt{m}\,\lceil r/2\rceil)$ to eliminate decryption errors because then, the accumulated error in $\mathbf{c}$ is distributed as a Gaussian with parameter $1/30$, which limits it to at most $1/6$ per component with high probability by a Gaussian tail bound. See, e.g., [Pei09], for the details. For simplicity, we choose $r = 2$. Notice that other trade offs, e.g., choosing a different (non-binary) message alphabet or choosing a larger r, are possible and easy to implement. Furthermore, one could choose a larger α and employ an error correcting code to eliminate decryption errors (cf. [MR08]).

We let $q = q(n)$ be the smallest prime between $2n^2$ and $4n^2$, Then, we set $m = m(n) = \lceil((n+\kappa)\log_2(q) + 2\kappa)/\log_2(r+1)\rceil$ to tie the probability of being able to distinguish ciphertexts from uniform to the symmetric security level, i.e., the probability is at most $\sqrt{q^{n+\kappa}/(r+1)^m} \leq \sqrt{q^{n+\kappa}/(q^{n+\kappa}2^{2\kappa})} = 2^{-\kappa}$. After taking all this into account, we propose various parameter sets in Table 3.4. There, we also include the resulting object sizes in kilobytes.

	year	2018	2010	2020	2030	2040	2050	2060	2070	2080	2090
	κ	128	150	164	176	190	204	216	230	244	256
Lenstra	**n**	214	191	221	253	283	314	346	376	405	438
	q	91621	72973	97687	128021	160183	197203	239441	282767	328051	383693
	α	5.47e-04	5.51e-04	5.12e-04	4.80e-04	4.54e-04	4.30e-04	4.10e-04	3.92e-04	3.77e-04	3.63e-04
	m	3719	3665	4234	4815	5400	6006	6609	7215	7811	8446
	$\lvert sk \rvert$	55.1	56.5	73.3	92.2	113.5	137.5	163	191.2	221	253.9
	$\lvert pk \rvert$	54.8	63.6	80.3	98	118.7	141.7	165.1	192	220.6	250.3
	$\lvert C \rvert$	0.7	0.7	0.8	0.9	1	1.1	1.2	1.3	1.5	1.6

The columns correspond to security until a given year. pk is the public key, sk is the secret key, and C is the ciphertext. All sizes are in kilobytes (kB).

Table 3.4: Recommended parameters for multi-bit LWE.

3.4 Conclusion and Open Problems

With our framework to analyze the SIS and LWE problems, we have established a connection between lattice problems and symmetric "bit security" for the first time. While our analysis reveals certain weaknesses in the way parameters for lattice-based cryptosystems are currently proposed, it also provides the tools to systematically do so for a sliding scale of security.

We propose that the presented methodology should be used whenever a new cryptographic primitive is presented to ensure that, concerning efficiency and security, it actually presents an improvement over known work. Furthermore, our work can be used to compare the security levels of parameter sets for entirely different cryptographic primitives, e.g., encryption and signature schemes. This is important when both are used in a more complex protocol, where all components should provide approximately the same level of security.

An additional application of our work is the proposal of parameters for lattice-based signature and encryption schemes for which there were no previously known concrete parameter sets. Doing so reveals that all schemes that require trapdoor matrices (cf. Section 2.7.2) are far from practical and seem to require an enormous effort to become so. On the other hand, such an analysis reveals that there are quite competitive signature and CPA encryption schemes already, especially those working in ideal lattices [Lyu09, LPR10].

To conclude, with our work we would like to draw renewed interest to the development of practical, strong lattice basis reduction algorithms for large dimensions as well as to further optimizing the parameter constraints for known lattice-based cryptosystems, which have mainly been of theoretic interest so far. An interesting extensions of our framework would be to also model and include the expected run-

ning time of security reductions from SIS and LWE, i.e., to take the tightness of the reductions into account. Another challenging direction would be to estimate the exact cost of combinatorial attacks, especially against lattice problems in the infinity norm.

Chapter 4

Blind Signatures

Since Chaum proposed his idea of blind signatures [Cha82], it has become an important primitive for anonymous Internet banking [Cha82, BBC$^+$94, AB09], e-voting [OMA$^+$99, IKSA03, RHOAGZ07], as well as for anonymous credential systems [Bra99, CG08, BP10].

The security model, mainly influenced by Juels, Luby, and Ostrovsky [JLO97] as well as Pointcheval and Stern [PS00], requires blind signature schemes to satisfy blindness and one-more unforgeability. Blindness states that the signer must not obtain any information on the signed messages and one-more unforgeability means that an adversary cannot obtain more signatures than there were interactions with the signer.

There are instantiations from general assumptions, e.g., by Juels et al. [JLO97], Fischlin [Fis06], Hazay, Katz, Koo, and Lindell [HKKL07], or Abe and Ohkubo [AO09]. The resulting instantiations are typically very inefficient, which is why we are interested in direct constructions.

All previous direct constructions, such as [Cha82, PS97, PS00, Abe01, BNPS03, CKW04, Oka06], have one thing in common: they are built upon classic number theoretic assumptions, like the hardness of factoring large integers or computing discrete logarithms. The more recent approaches, e.g., by Boldyreva [Bol03] or Okamoto [Oka06], tend to use pairings that yield very elegant constructions. They, however, are again based on the discrete logarithm problem in this specific setting.

Our Contribution We construct the first lattice-based blind signature scheme. It is inspired by Lyubashevsky's identification scheme [Lyu08a, Lyu08b] in combination with the Fiat-Shamir paradigm [FS86]. It is unconditionally blind, selective-failure blind [CNS07], and one-more unforgeable in the random oracle model under an SIS hardness assumption. We summarize the main results for our scheme BS.

Theorem (Blindness)**.** *BS is blind in a statistical sense.*

Theorem (One-more unforgeability)**.** *Let* $D \subseteq \mathbf{R}_0$ *such that* $\mathbf{f} \in D$ *if* $\|\mathbf{f}\|_\infty = \widetilde{\mathcal{O}}(n^4)$*. BS is one-more unforgeable in the random oracle model if* $COL(\mathcal{H}(\mathbf{R}, m), D)$ *is hard against sub-exponential attacks.*

With its four moves it is round-efficient. All operations have quasi-linear complexity and all keys and signatures require a quasi-linear amount of storage bits with respect to the main parameter n. Moreover, it is the first leakage resilient blind signature scheme. Our model of leakage resilience is inspired by Katz and Vaikuntanathan [KV09]. Let L be the bit-length of the secret key. Our scheme remains secure, even if the adversary obtains $L(1 - o(1))$ bits of the secret key via arbitrary

side channels. This brings the security model closer to reality, where the adversary may obtain information about the secret key, e.g, via (remote) timing attacks or by having physical access to the signing device. Such a resilience also improves the trust that we are willing to grant these schemes.

Table 4.1 compares RSA and Okamoto-Schnorr (OS) [PS00] blind signatures with our construction in terms of computational cost. For all schemes, we propose parameter sets for current, medium, and future security levels. We believe that RSA is a good basis for comparison because it is easy to understand and very efficient as blind signing only involves two modular exponentiations and verification can be done in a single one. We do *not* count multiplications. As observed in [BNPS03], the security of the RSA blind signature scheme is based on a specially tailored interactive assumption that is stronger than the original RSA assumption [BMV08]. Taking all this into account, the timings observed for RSA provide an optimistic lower bound for current practical and secure schemes. The timings for OS are expected timings based on the number of modular exponentiations, *not* counting multiplications. We include OS because it follows the typical 3-move structure and is based on standard assumptions. It is therefore closer to our protocol. The timings were obtained on an AMD Opteron CPU, running at 2.3 GHz. For RSA and OS, we have used OpenSSL 0.9.8g, which is supposed to be very efficient. For our blind signature schemes, we did a straightforward implementation, which certainly leaves room for improvements. Here, the timings reflect the *full* scheme.

From Table 4.1, we clearly see that our scheme benefits from its quasi-linear complexity, especially in higher levels of security. In addition, for our scheme, we can have various trade-offs between signature size and speed. For more details, refer to Section 4.3. There, we also show how to optimize the key and signature sizes.

Main Obstacles For every blind signature scheme, one has to overcome three basic obstacles. The scheme needs to be blind, one-more unforgeable, and at the same time complete. Blindness and unforgeability are already somewhat orthogonal because granting the user too much power to ensure blindness harms unforgeability and vice-versa. Since working with lattices, we do not have access to a cyclic group structure as in schemes that are based on the DDH or DL assumptions. There, blindness is typically easier to achieve by multiplying the message with a random group element.

In lattices, we need to emulate this over an infinite structure via a filtering technique that is inspired by [Lyu08a]. However, this technique introduces a completeness defect that even affects the interaction of an honest user with an honest signer. Thus, the protocol may need to be restarted. We show how this technique can be

Scheme	Secure until	Security (bits)	Moves	KeyGen	Sign	Verify
RSA-1229	2012	Current (76)	2	95 ms	16 ms	5 ms
RSA-3313	2050	Medium (102)	2	1250 ms	46 ms	6 ms
RSA-15424	2282	Future (256)	2	251849 ms	2134 ms	20 ms
OS-1229	2012	Current (76)	3	16 ms	64 ms	24 ms
OS-3313	2050	Medium (102)	3	46 ms	184 ms	69 ms
OS-15424	2282	Future (256)	3	2134 ms	8536 ms	3201 ms
Section 4.2 ($n = 1024$)	2012	Current (76)	4	37 ms	220 ms	33 ms
Section 4.2 ($n = 2048$)	2050	Medium (102)	4	52 ms	283 ms	57 ms
Section 4.2 ($n = 8192$)	2282	Future (256)	4	305 ms	1175 ms	320 ms

The table compares our scheme with RSA and Okamoto-Schnorr for various moduli according to [Len05] (Current, Medium) and [ECR10] (Future). The bitlengths can be computed on `www.keylength.com`. For our blind signature scheme, we propose three *optimized* parameter sets for the same security levels based on Chapter 3. Note that the parameters for RSA and OS do *not* take potential quantum-computer attacks into account. All timings are averaged over 1000 random instances.

Table 4.1: Comparison of RSA, Okamoto-Schnorr, and our blind signature scheme.

refined to allow a time-memory trade-off, reducing the number of expected restarts at the expense of only slightly larger signatures. When addressing this defect, we need additional means to ensure blindness over repetitions of the protocol. Our solution involves a statistically hiding commitment.

Similarly, the completeness defect has implications with respect to unforgeability as the user may claim that the protocol has failed, whereas it was indeed successful. Here, we extend the typical three-move structure to a four-move structure where the user needs to demonstrate that he or she could not obtain a valid signature. Such a last move, from user to signer, is highly unusual for blind signature schemes. We solve this issue by designing a special proof of failure and by employing a computationally binding commitment scheme.

RSA-style Blind Signatures One might think that RSA-style (hash $\rightarrow$ blind $\rightarrow$ invert $\rightarrow$ unblind) lattice-based blind signatures can be implemented using the preimage sampleable trapdoor function family PSTF. Let $f : \mathbb{Z}^m \rightarrow \mathbb{Z}_q^n$ be such a function. The user would hash the message msg using a full-domain hash random oracle $h \leftarrow \mathsf{H}(msg)$ and blind using $msg^* \leftarrow h + f(\beta)$ for $\beta \leftarrow \mathsf{SampleDom}(m, \eta)$. The signer would *sample* from $f^{-1}(msg^*)$ and return a short vector σ^*. Using β and

the fact that f is linear, the user can compute $\sigma \leftarrow \sigma^* - \beta$, which is short and passes verification, $f(\sigma) \equiv f(\sigma^*) - f(\beta) \equiv \mathsf{H}(msg^*)$. For the proof, one would rely on an interactive "one-more" trapdoor inversion assumption akin to [BNPS03]. However, the adversary must never obtain a short non-zero $\mathbf{x}$ such that $f(\mathbf{x}) = \mathbf{0}$ because this would imply learning a piece of a short basis. Unfortunately, such an attack is easy: take $u \leftarrow \mathsf{SampleDom}(m, \eta)$ and send $msg^* = f(\mathbf{u})$ to the signer, who returns σ^*. Now, $\mathbf{x} \leftarrow \mathbf{u} - \sigma^*$ is short and $f(\mathbf{x}) = \mathbf{0}$. Also, $\mathbf{x} \neq \mathbf{0}$ with high probability.

Organization After a brief discussion of the security model and potential extensions in Section 4.1. we propose our blind signature scheme in Section 4.2 and prove its security. Section 4.3 is devoted to selecting secure parameters based on Chapter 3 that yield an efficient blind signature scheme. Before we conclude the chapter in Section 4.5, we prove several supporting lemmas in Section 4.4.

This chapter is a reprint of [Rüc10b]. The dissertation author was the primary investigator and author of this paper.

4.1 Definitions

A blind signature scheme BS consists of three algorithms $(\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$, where Sign is an interactive protocol between a signer $\mathcal{S}$ and a user $\mathcal{U}$. The specification is as follows.

Key Generation: $\mathsf{Kg}(1^n)$ outputs a private signing key sk and a public verification key pk.

Signature Protocol: $\mathsf{Sign}(sk, msg)$ describes the joint execution of $\mathcal{S}$ and $\mathcal{U}$. The private output of $\mathcal{S}$ is a view V and the private output of $\mathcal{U}$ is a signature σ on the message $msg \in \mathcal{M}$ with message space $\mathcal{M}$ under sk. Thus, we write $(V, \sigma) \leftarrow \langle \mathcal{S}, \mathcal{U} \rangle (sk, (pk, msg))$.

Signature Verification: $\mathsf{Vf}(pk, \sigma, msg)$ outputs 1 if σ is a valid signature for msg under pk; otherwise 0.

Completeness is defined as in digital signature schemes, i.e., every honestly created signature for honestly created keys and for any message $msg \in \mathcal{M}$ has to be valid under this key. Views are interpreted as random variables, whose output is generated by subsequent executions of the respective protocol. Two views V_1 and V_2 are considered equal if they cannot be distinguished in a statistical sense.

As for security, blind signatures have to satisfy two properties: blindness and one-more unforgeability [JLO97, PS00]. The notion of blindness is defined in the experiment $\mathsf{Exp}^{\mathsf{blind}}_{\mathcal{S},\mathrm{BS}}$ in Figure 4.1, where the adversarial signer $\mathcal{S}$ works in three

Experiment $\mathsf{Exp}^{\mathsf{blind}}_{\mathcal{S},\mathrm{BS}}(n)$
$b \leftarrow_{\$} \{0,1\}$
$(pk, sk) \leftarrow \mathrm{BS}.\mathsf{Kg}(1^n)$
$(msg_0, msg_1, s\_find) \leftarrow \mathcal{S}(m\_find, sk, pk)$
$s\_issue \leftarrow \mathcal{S}^{\langle\cdot,\mathcal{U}(pk,msg_b)\rangle^1,\langle\cdot,\mathcal{U}(pk,msg_{1-b})\rangle^1}(m\_issue, s\_find)$
Let σ_b and σ_{1-b} be the outputs of $\mathcal{U}(pk, msg_b)$
and $\mathcal{U}(pk, msg_{1-b})$, respectively.
If $\sigma_0 \neq fail$ and $\sigma_1 \neq fail$
  $d \leftarrow \mathcal{S}(m\_guess, s\_issue, \sigma_0, \sigma_1)$
Else
  $d \leftarrow \mathcal{S}(m\_guess, s\_issue, fail, fail)$
Return 1 iff $d = b$

Experiment $\mathsf{Exp}^{\mathsf{omf}}_{\mathcal{U}^*,\mathrm{BS}}(n)$
$(pk, sk) \leftarrow_{\$} \mathrm{BS}.\mathsf{Kg}(1^n)$
$\{(msg_1, \sigma_1), \ldots, (msg_\jmath, \sigma_\jmath)\} \leftarrow \mathcal{U}^{*\langle\mathcal{S}(sk),\cdot\rangle^\infty}(pk)$
Let ℓ be the number of successful interaction
between $\mathcal{U}^*$ and the signer.
Return 1 iff
  1. $msg_i \neq msg_j$ for all $1 \leq i < j \leq \jmath$;
  2. $\mathrm{BS}.\mathsf{Vf}(pk, \sigma_i, msg_i) = 1$ for all $i = 1, \ldots, \jmath$;
  3. $\ell + 1 = \jmath$.

Figure 4.1: Security experiments for blindness and one-more unforgeability of blind signatures.

modes.

In mode *find*, it chooses two messages msg_0, msg_1 and interacts with two users in mode *issue*. Depending on a coin flip b, the first (second) user obtains a blind signature for msg_b (msg_{1-b}). After seeing the unblinded signatures in the original order, with respect to msg_0, msg_1, the signer has to guess the bit b in mode *guess*. If either of the user algorithms fails in outputting a valid signature, the signer is merely notified of the failure and does not get any signature. Below, we deal with aborts as an extension. Also note that we allow the adversary to keep a state that is fed back in subsequent calls.

A scheme BS is (t, δ)-blind, if there is no adversary $\mathcal{S}$, running in time at most t, that wins the above experiment with advantage more than δ, where the advantage is defined as $\mathsf{Adv}^{\mathsf{blind}}_{\mathcal{S},\mathrm{BS}} = \left|\mathrm{Prob}\left[\mathsf{Exp}^{\mathsf{blind}}_{\mathcal{S},\mathrm{BS}}(n) = 1\right] - \frac{1}{2}\right|$. A scheme is *statistically* blind if it is (∞, δ)-blind for a negligible δ. The second security property, one-more unforgeability, ensures that each completed interaction between signer and user yields at most one signature. It is formalized in the experiment $\mathsf{Exp}^{\mathsf{omf}}_{\mathcal{U}^*,\mathrm{BS}}$, where an adversarial user tries to output $\jmath$ valid signatures after $\ell < \jmath$ completed interactions with an honest signer. $\mathcal{H}$ is a family of random oracles.

A signature scheme BS is $(t, Q_{\mathsf{OSign}}, \delta)$-one-more unforgeable if there is no adversary $\mathcal{A}$, running in time at most t, making at most Q_{OSign} signature queries, that wins the above game with probability more than δ.

4.1.1 Extensions

We consider three extensions to the above security model for blind signatures: one deals with user aborts, the second with dishonestly chosen keys, and the third with

leakage resilience.

Security Under Aborts Blindness in the previous subsection does not cover the case where the protocol is aborted prematurely. There is the strengthened notion of selective failure blindness [CNS07], where the malicious signer may choose either msg_0 or msg_1 according to some secret distribution that makes the protocol fail. Preventing this generically is easy as was shown by Fischlin and Schröder in [FS09]. In the course of the discussion of our construction, we argue that it already is blind in this sense.

Adversely-chosen Keys Consider the blindness experiment in [ANN06]. Instead of having the experiment select pk, sk, we can let the signer output pk. Blindness may be harder to achieve in this setting. However, our construction remains blind in this stronger model as the proof does not exploit specifics about the key.

Leakage Resilience Resilience to key leakage is a way to ensure security against side-channel attacks.

In [KV09], Katz and Vaikuntanathan give a nice overview of past developments and the evolution of leakage resilience for authenticity [ADW09, KV09] and secrecy, e.g., [DP08, AGV09, NS09]. Obviously, we are interested in authenticity in the special case of blind signatures. We model key leakage in the new unforgeability experiment $\mathsf{Exp}^{\mathsf{omf},\lambda\text{-}\mathsf{OLeak}}_{\mathcal{U}^*,\mathrm{BS}}$ by adding a leakage oracle $\mathsf{OLeak}(\cdot)$ to $\mathsf{Exp}^{\mathsf{omf}}_{\mathcal{U}^*,\mathrm{BS}}$. The adversary can adaptively query OLeak with a series of functions f_i, $i \in \{1, \ldots, \kappa\}$, and receives $f_i(sk)$. The only restriction is that $\sum_{i=1}^{\kappa} |f_i(sk)| \leq \lambda(|sk|)$, where the function λ determines the amount of leakage that we are willing to tolerate. Notice that the signer's key does not have to evolve over time and its secret state consists of the secret key only. Furthermore, observe that this extension is only sensible as long as $\lambda(|sk|) < \min\{|sk|, |\sigma|\}$. Otherwise, the adversary could easily obtain the entire secret key or a signature of its choice. The scheme BS is leakage-resilient with λ if there is no efficient adversary $\mathcal{U}^*$ for which the experiment outputs 1. To demonstrate leakage resilience, one has to show that the conditional min-entropy $H_\infty(sk|\mathsf{OLeak}(sk)) := \min_{sk'}\{-\log(\mathrm{Prob}\left[\, sk = sk' | \mathsf{OLeak}(sk)\right])\}$ of the secret key is still sufficiently large to prove security. In our case, we need to ensure witness-indistinguishability of the underlying proof of knowledge, even under leakage, and we need to be able to answer the adversary's queries to the leakage oracle.

Experiment $\mathsf{Exp}^{\mathsf{omf},\lambda\text{-}\mathsf{OLeak}}_{\mathcal{U}^*,\mathrm{BS}}(n)$
$(pk, sk) \leftarrow \mathrm{BS}.\mathsf{Kg}(1^n)$
$\{(msg_1, \sigma_1), \ldots, (msg_\jmath, \sigma_\jmath)\} \leftarrow \mathcal{U}^{*\langle \mathcal{S}(sk), \cdot \rangle^\infty, \mathsf{OLeak}(sk, \cdot)}(pk)$
Let ℓ be the number of successful interaction between $\mathcal{U}^*$ and the signer.
Let $f_1, \ldots, f_\kappa$ be the leakage queries of $\mathcal{U}^*$, each with output length λ_i.
Return 1 iff
1. $msg_i \neq msg_j$ for all $1 \leq i < j \leq \jmath$;
2. $\mathrm{BS}.\mathsf{Vf}(pk, \sigma_i, msg_i) = 1$ for all $i = 1, \ldots, \jmath$;
3. $\ell + 1 = \jmath$;
4. $\sum_{i=1}^{\kappa} \lambda_i \leq \lambda(|sk|)$.

4.1.2 Commitments

Commitments typically work in two phases. First, one party publishes a commitment $C = \mathsf{com}(msg; r) \in \{0,1\}^n$, $r \leftarrow_\$ \{0,1\}^n$, to a message $msg \in \{0,1\}^*$ without revealing any information about it. This is the "hiding" property of the commitment scheme. In the second phase, the party can prove that C actually corresponds to msg by revealing r. It is important that no efficient algorithm can find a second message msg' and randomness r' such that $C = \mathsf{com}(msg'; r')$, i.e., break the "binding" property. As usual, these properties are defined for families of such commitment functions. A scheme is (t, δ)-hiding (-binding) if there is no algorithm running in time at most t that can break the hiding (binding) property with probability more than δ. Both properties can be satisfied computationally or unconditionally but there is no scheme that is unconditionally hiding *and* unconditionally binding [Gol04].

As we are interested in fully lattice-based schemes, we would like to point out that such commitment schemes can be built upon hard lattice problems [KTX08]. In practice, one rather uses cryptographic hash functions, such as [ADL+08], in a special mode.

In particular, for our scheme, we will require a statistically $\delta^{(h)}_{\mathsf{com}}$-hiding and computationally $(t_{\mathsf{com}}, \delta^{(b)}_{\mathsf{com}})$-binding commitment scheme.

4.2 Our Construction

We construct a lattice-based blind signature scheme. It is secure in the random oracle model under a worst-case assumption in ideal lattices and its time *and* space complexity is quasi-optimal, $\widetilde{\mathcal{O}}(n)$.

Parameter	Value	Asymptotics	Usage
d_s	positive integer constant $< q/(8n)$	$\mathcal{O}(1)$	secret key size, unforgeability
D_s	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d_s\}$		set of secret keys
c_m	$> 1/\log(2d_s)$	$\widetilde{\mathcal{O}}(1)$	witness indistinguishability, leakage resilience
m	$\lfloor c_m \log(q) \rfloor + 1$	$\Omega(\log(n))$	worst-case to average-case reduction
D_ϵ	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq 1 =: d_\epsilon\}$	$\mathcal{O}(1)$	hash output size
ϕ, ψ	positive integer constant ≥ 1	$\mathcal{O}(1)$	completeness, speed
D_α	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq \psi n d_\epsilon =: d_\alpha\}$	$\mathcal{O}(n)$	blindness
D_{ϵ^*}	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d_\alpha - d_\epsilon =: d_{\epsilon^*}\}$	$\mathcal{O}(n)$	blindness
D_y	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq \phi m n^2 d_s d_{\epsilon^*} =: d_y\}$	$\widetilde{\mathcal{O}}(n^3)$	witness indistinguishability
G_*	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d_y - n d_s d_{\epsilon^*} =: d_{G_*}\}$	$\widetilde{\mathcal{O}}(n^3)$	witness indistinguishability, completeness defect
D_β	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq \phi m n d_{G_*} =: d_\beta\}$	$\widetilde{\mathcal{O}}(n^4)$	blindness
G	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d_\beta - d_{G_*} =: d_G\}$	$\widetilde{\mathcal{O}}(n^4)$	blindness, completeness defect
D	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d_{G_*} + d_\beta + n d_s d_\epsilon =: d_D\}$	$\widetilde{\mathcal{O}}(n^4)$	collisions under h
q	$\geq 4mn\sqrt{n}\log(n) d_D$, prime	$\widetilde{\Theta}(n^5\sqrt{n})$	worst-case to average-case reduction

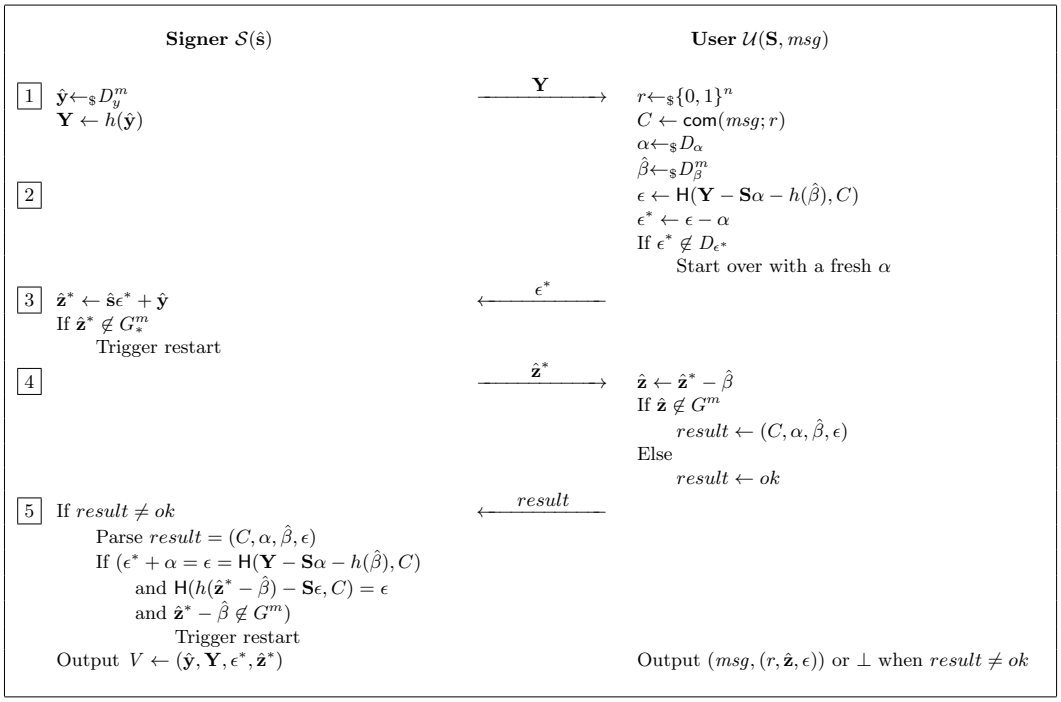
The table defines all parameters and sets for our scheme. The sets are defined via a norm bound, for which we also state the asymptotic growth with respect to n. The last column states the main usage for the individual parameter or set. All sets are subsets of the ring $\mathbf{R}_0 = \mathbb{Z}[X]/(X^n + 1)$.

Table 4.2: Parameters for the security parameter n.

The road map for this section is as follows: We describe the 4-move blind signature scheme BS. Then, we prove completeness, blindness, and one-more unforgeability. Proving completeness is non-trivial as we need to address an inevitable completeness defect. In the course of the discussion we show that it neither harms security nor efficiency. Afterwards, we prove that the scheme is statistically blind, one-more unforgeable, and leakage-resilient.

The scheme requires a large number of parameters that need to be carefully worked out. Their definition in Table 4.2 will be justified later in the analysis. We chose not to "unwind" the parameters d_s, d_ϵ, etc., because we need their relative size in the various lemmas below, making the proofs easier to understand. The asymptotics in the third column should help estimating their magnitude. The parameter d_ϵ is a constant 1 here but it can be increased if it is necessary to sign hash values of bit length $> n \log_2(3)$. The "usage" hint in the table points at the section, where they are most influential.

As for selecting practical parameters, we refer the reader to Section 4.3. There, we propose secure parameter sets based on the analysis in Chapter 3.



All parameters and sets are defined in Table 4.2. Note that the signer implicitly verifies that the user's protocol messages come from the correct domains.

Figure 4.2: Issue protocol of the blind signature scheme BS.

4.2.1 Informal Description

We give a detailed, slightly informal description of the protocol Steps 1-5 in Figure 4.2 and of the influence of the parameter relations in Table 4.2.

Basically, the protocol follows the structure of a (canonical) 3-move identification scheme, which provides a witness-indistinguishable proof of knowledge (cf. Section 2.5). The signer proves knowledge of $\hat{\mathbf{s}} \in D_s^m$ such that $h(\hat{\mathbf{s}}) = \mathbf{S}$ with $\mathbf{S}$ being the public key. The function h is chosen from the family $\mathcal{H}(\mathbf{R}, m)$.

We stick to this basic structure and let the signer transmit a "commitment" $\mathbf{Y} = h(\hat{\mathbf{y}})$ for a random value $\hat{\mathbf{y}} \in D_y^m$. The user computes the blinded message ϵ^* as a function (involving H) of $\mathbf{Y}$ and the to-be-signed message msg and sends it to the signer, which returns $\hat{\mathbf{z}}^* = \hat{\mathbf{s}}\epsilon^* + \hat{\mathbf{y}}$. Via the linearity of h, the user can verify that $h(\hat{\mathbf{z}}^*) = \mathbf{S}\epsilon^* + \mathbf{Y}$ using only public knowledge. Afterwards, the user transforms

the blinded signature $(\hat{\mathbf{z}}^*, \epsilon^*)$ into the regular signature $(\hat{\mathbf{z}}, \epsilon)$ for *msg*.

However, to obtain a blind signature scheme from this strategy, we need to overcome three main obstacles. First, the scheme needs to be unforgeable, i.e., neither $\mathbf{Y}$ nor $\hat{\mathbf{z}}^*$ may leak too much information about the secret key. Second, ϵ^* and $\hat{\mathbf{z}}$ need to be distributed independently of the message *msg* to ensure blindness. Third, we need to ensure completeness despite the following, imperfect filtering technique.

Roughly speaking, we add two numbers $a \in [-A, A]$ and $b \leftarrow_\$ [-3A, 3A]$ and filter the output in the sense that we only reveal $c = a + b$ if $c \in [-2A, 2A]$. Otherwise, we choose a fresh b and try again. This ensures that c is distributed independently of a. However, the filtering technique in this example only works with probability $\approx 2/3$ and we expect $c \in [-2A, 2A]$ after $\approx 3/2$ trials.

It is applied to $(\hat{\mathbf{s}}\epsilon^*) + \hat{\mathbf{y}}$ to hide the secret key $\hat{\mathbf{s}}$ by randomly choosing the coefficients of $\hat{\mathbf{y}}$ from a relatively large set, compared to $\|\hat{\mathbf{s}}\epsilon^*\|_\infty$, and then filtering the output until it is in G_*^m. Whenever filtering fails in Step 3, the signer has to restart the entire protocol. After a small number of restarts, the signer can safely send $\hat{\mathbf{z}}^*$ without revealing the secret key $\hat{\mathbf{s}}$. Actually, the filtering technique is more involved and we need to deal with sums of vectors. We will show the details and a refinement in the next section.

Interestingly, the filtering technique can also be applied to achieve blindness. For the protocol message $\epsilon^* = \epsilon - \alpha$ after Step 2, we use $\alpha \leftarrow_\$ D_\alpha^m$ to hide ϵ. This is necessary, as ϵ will be a part of the output signature. The completeness defect in this filtering step can be eliminated because the user can repeat this step locally. In Step 4, the user attempts to unblind the signature by computing $\hat{\mathbf{z}} \leftarrow \hat{\mathbf{z}}^* - \hat{\beta}$, where $\hat{\beta}$ is prepared in Step 1 and randomly chosen from a relatively large set to hide $\hat{\mathbf{z}}^*$. This is the third application of the filtering technique. If it fails, the protocol needs to be restarted.

This last defect is the reason for having the last move and Step 5. Even if both parties are honest, the user might not be able to obtain a valid signature with non-negligible probability in Step 4. This is highly unusual for a blind signature scheme, in fact we are not aware of any other schemes with this kind of behavior.[1] In such a case, the user needs to prove that no valid signature could be obtained (Step 5) and the protocol needs to be restarted. Therefore, the user submits $(C, \alpha, \hat{\beta}, \epsilon)$ to the signer, where C is a commitment to the message *msg*. The signer can verify that the user was unable to obtain a valid signature relative to the commitment C.

[1] However, existing schemes may benefit from introducing such a defect. For example, [Lyu09] improves the efficiency of a (regular) signature scheme based on the factoring problem by introducing a similar defect.

For unforgeability, we require that the commitment is binding and for blindness it is crucial that it is hiding. The hiding property of the commitment also prevents the signer from learning information about *msg* across restarts.

Since restarts do not harm security, we can repeat the protocol until it is complete. The expected number of repetitions is constant and it can be brought close to 1 by choosing the parameters appropriately.

The above may sound like a generic transformation to achieve perfect completeness for any blind signature scheme with a defect. In fact, it requires a computational assumption that is specifically tailored to the scheme. Still, we believe that the general technique can be easily transferred to other schemes.

4.2.2 Our Construction

We construct our blind signature scheme BS = $(\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ as follows.

Key Generation: $\mathsf{Kg}(1^n)$ selects a secret key $\hat{\mathbf{s}} \leftarrow_\$ D_s^m$, and a compression function $h \leftarrow_\$ \mathcal{H}(\mathbf{R}, m)$. Let $\mathsf{H} : \mathbf{R} \times \{0,1\}^n \to D_\epsilon$ be a random oracle and let $\mathcal{C}(1^n)$ be a commitment scheme, mapping $\{0,1\}^* \times \{0,1\}^n \to \{0,1\}^n$. The algorithm chooses a function $\mathsf{com} \leftarrow_\$ \mathcal{C}(1^n)$.

Then, it computes the public key $\mathbf{S} \leftarrow h(\hat{\mathbf{s}})$ and outputs $(\hat{\mathbf{s}}, \mathbf{S})$. For simplicity, we treat h, com, and the parameters in Table 4.2 as globally known and implicit inputs to all algorithms. However, each signer may choose them individually and include them in the public key.

Signature Protocol: The signature issue protocol for messages $msg \in \{0,1\}^*$ is depicted in Figure 4.2. Eventually, the user outputs a message *msg* and a signature $(r, \hat{\mathbf{z}}, \epsilon)$.
Notes: Upon a restart after Step 2, the user only selects a fresh $\alpha \leftarrow_\$ D_\alpha$ and repeats the operations that involve α. Whenever the signer triggers a restart, the user chooses a fresh r in order to make the protocol execution independent of the previous ones. Therefore, we omit values from previous runs in the signer's view. During Step 5, the signer can detect a cheating user that tries to trigger a restart, despite having received a valid signature. In this case, the signer can stop the protocol and assume that the user has obtained a valid signature.

Verification: $\mathsf{Vf}(\mathbf{S}, (r, \hat{\mathbf{z}}, \epsilon), msg)$ outputs 1 if $\hat{\mathbf{z}} \in G^m$ as well as $\mathsf{H}(h(\hat{\mathbf{z}}) - \mathbf{S}\epsilon, \mathsf{com}(msg; r)) = \epsilon$; otherwise 0.

4.2.3 Analysis and Security

In this section, we analyze our blind signature scheme with regard to completeness, blindness, one-more unforgeability, and leakage resilience. For each aspect, we prove a main theorem. Supporting lemmas are stated before the theorems and they are proven in Section 4.4.

Completeness

Completeness of BS is a non-trivial issue due to the eventual restarts and the many parameters involved. The next lemma ensures that the number of restarts is small, effectively constant.

Lemma 4.1. *Let* $k = \Omega(n)$, $A, B \in \mathbb{N}_{>0}$, $\mathbf{a}, \mathbf{b} \in \mathbb{Z}^k$ *with* arbitrary $\mathbf{a} \in \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq A\}$ *and* random $\mathbf{b} \leftarrow_\$ \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq B\}$. *Given* $B \geq \phi k A$ *for* $\phi \in \mathbb{N}_{>0}$, *we have* $\Prob_{\mathbf{b}}[\,\|\mathbf{a} - \mathbf{b}\|_\infty \leq B - A] > e^{-1/\phi} - o(1)$.

The multiplication of two polynomials modulo $X^n + 1$ plays a major role in the analysis. Therefore, we need the following lemma, which is a special case of [Lyu08b, Lemma 2.8].

Lemma 4.2. *For any two* $\mathbf{a}, \mathbf{b} \in \mathbb{Z}[X]$ *of degree* n, *we have* $\|\mathbf{ab} \bmod (X^n + 1)\|_\infty \leq n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$.

Theorem 4.3 (Completeness)**.** *Let* $g(n) = \omega(\log^2(n))$. *The scheme BS is complete after at most* $g(n)$ *(or, an expected number of* $e^{2/\phi}$*) repetitions.*

Proof. Let us assume that the protocol does not have to be restarted after Steps 2, 3, and 4. Then, for all honestly generated key pairs $(\hat{\mathbf{s}}, \mathbf{S})$, all messages $msg \in \{0,1\}^*$, and all signatures $(r, \hat{\mathbf{z}}, \epsilon)$ we have $\hat{\mathbf{z}} \in G^m$ and $h(\hat{\mathbf{z}}) - \mathbf{S}\epsilon = h(\hat{\mathbf{z}}^* - \hat{\beta}) - \mathbf{S}\epsilon = h(\hat{\mathbf{s}}(\epsilon - \alpha) + \hat{\mathbf{y}} - \hat{\beta}) - \mathbf{S}\epsilon = \mathbf{Y} - \mathbf{S}\alpha - h(\hat{\beta})$ and $\mathsf{com}(msg; r) = C$. Therefore $\mathsf{H}(h(\hat{\mathbf{z}}) - \mathbf{S}\epsilon, \mathsf{com}(msg; r)) = \epsilon$ and $\mathrm{BS.Vf}(\mathbf{S}, (r, \hat{\mathbf{z}}, \epsilon), msg) = 1$.

Now, we analyze the probability of a restart. Observe that the restarts after Step 2 do not affect completeness, as the user does them locally. The number of trials here is at most $g(n)$ for any $g(n) = \omega(\log(n))$ due to Lemma 4.1 ($k = n, A = d_s, B = d_\alpha$) for $\epsilon - \alpha \in D_{\epsilon^*}$. However, the expected number of trials is constant ($e^{1/\psi}$). It is safe to set $\psi = 1$ here but one might want less trials, e.g., less than 1.5 for $\psi \geq 3$ and $n > 1$.

After Steps 3 and 4, aborts affect the protocol and trigger a full restart. In Step 3, we need to ensure that $\hat{\mathbf{s}}\epsilon^* + \hat{\mathbf{y}} \in G_*^m$. By Lemma 4.2, we know that

$\|\hat{\mathbf{s}}\epsilon^* \bmod (X^n+1)\|_\infty \leq nd_s d_{\epsilon^*}$ and applying Lemma 4.1 ($k = mn, A = nd_s d_{\epsilon^*}, B = d_y$) yields the constant success probability $e^{-1/\phi}$ and a maximum number $g(n)$ of trials for any $g(n) = \omega(\log(n))$. In practice, this can be optimized by increasing ϕ. After an expected number $e^{1/\phi}$ of restarts, the protocol proceeds to Step 4.

In Step 4, the user attempts to unblind the signature and requires that $\hat{\mathbf{z}}^* - \hat{\beta} \in G^m$. Otherwise, the user convinces the signer that a restart is necessary. We apply Lemma 4.1 on ($k = mn, A = d_{G_*}, B = d_\beta$) and obtain the same behavior as in Step 3.

In total, after at most $g(n)$, for any $g(n) = \omega(\log^2(n))$, or an expected number $e^{2/\phi}$ of trials, the protocol is complete. □

In Section 4.3, we will see that $\phi = 4$ is good choice to make the protocol more efficient in practice. Observe that in any case, all operations (including eventual restarts) in BS have $\widetilde{\mathcal{O}}(n)$ complexity and that private keys, public keys, and signatures have size $\widetilde{\mathcal{O}}(n)$.

Blindness

We prove that BS is statistically blind based on the observation that the signer only sees values that are independent of the message being signed. More precisely, the views generated by two different messages are indistinguishable. For this argument to work, we require a statistically hiding commitment scheme and appropriate sets D_α, D_β, D_{ϵ^*}, and G. The following probabilistic lemma is crucial as it guarantees that the user's message after Step 2 and the final output are independent of the message.

Lemma 4.4. *Let* $k \in \mathbb{N}_{>0}$, $A, B \in \mathbb{N}_{>0}$, $\mathbf{a}, \mathbf{a}', \mathbf{b} \in \mathbb{Z}^k$ *with* arbitrary $\mathbf{a}, \mathbf{a}' \in \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq A\}$, *a* random $\mathbf{b} \leftarrow_\$ \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq B\}$ *for* $B > A$. *We define the random variables* $\mathbf{c} \leftarrow \mathbf{a} - \mathbf{b}$ *and* $\mathbf{c}' \leftarrow \mathbf{a}' - \mathbf{b}$ *if* $\max\{\|\mathbf{a} - \mathbf{b}\|_\infty, \|\mathbf{a}' - \mathbf{b}\|_\infty\} \leq B - A$, *otherwise, we resample* $\mathbf{b}$. *Then,* $\Delta(\mathbf{c}, \mathbf{c}') = 0$.

The role of com is to ensure that the signer can only obtain negligible information from restarts. Notice that BS is perfectly blind ($(\infty, 0)$-blind) if the commitment scheme is perfect (0-hiding).

Theorem 4.5 (Blindness)**.** *BS is* $(\infty, \delta^{(h)}_{\mathsf{com}})$*-blind if* com *is* $\delta^{(h)}_{\mathsf{com}}$*-hiding.*

Proof. As per experiment $\mathsf{Exp}^{\mathsf{blind}}_{\mathcal{S},\mathrm{BS}}$, the adversarial signer outputs two messages msg_0, msg_1 and interacts with two users $\mathcal{U}(\mathbf{S}, msg_b)$, $\mathcal{U}(\mathbf{S}, msg_{1-b})$ after a secret

coin flip $b \leftarrow \{0,1\}$. We show that these users do not leak any information about their respective message.

Technically, we establish that all protocol messages and the output, when interpreted as random variables, are distributed independently of the message being signed. This involves an analysis of ϵ^*, $\hat{\mathbf{z}}$, and eventual restarts. As for ϵ and r, we already know that they are chosen uniformly at random.

Distribution of ϵ^*: Let $\epsilon_b^*, \epsilon_{1-b}^*$ be the first protocol messages of $\mathcal{U}(pk, msg_b)$ and $\mathcal{U}(pk, msg_{1-b})$, respectively. They are in D_{ϵ^*} and they are both of the form $\epsilon - \alpha$ with $\epsilon \in D_\epsilon$ and $\alpha \leftarrow_\$ D_\alpha$. The statistical distance $\Delta(\epsilon_b^*, \epsilon_{1-b}^*)$ is 0 by Lemma 4.4 ($k = n, A = d_s, B = d_\alpha$) because they are filtered to be in D_{ϵ^*}, i.e., their coefficients are bounded by $B - A = d_\alpha - d_s$.

Distribution of $\hat{\mathbf{z}}$: Let $\hat{\mathbf{z}}_0, \hat{\mathbf{z}}_1$ be part of the final output of $\mathcal{U}(pk, msg_0)$ resp. $\mathcal{U}(pk, msg_1)$. Both are of the form $\hat{\mathbf{z}}^* - \hat{\beta}$ for $\hat{\mathbf{z}}^* \in G_*^m$ and $\hat{\beta} \leftarrow_\$ D_\beta^m$. Furthermore, $\hat{\mathbf{z}}_0$ and $\hat{\mathbf{z}}_1$ are filtered to be in G^m, having coefficients bounded by $d_\beta - d_{G_*}$. Hence, the statistical distance $\Delta(\hat{\mathbf{z}}_0, \hat{\mathbf{z}}_1)$ is 0 because of Lemma 4.4 ($k = mn, A = d_{G_*}, B = d_\beta$).

Restarts: Observe that each protocol run is statistically independent of the previous runs by the statistical hiding property of the commitment com and because the user selects fresh $r, \alpha, \hat{\beta}$ after every restart. This is the reason why we inherit the statistical $\delta_{\mathsf{com}}^{(h)}$-hiding property to obtain $(\infty, \delta_{\mathsf{com}}^{(h)})$-blindness instead of perfect blindness. Finally, we need to argue about the restart after Step 4. The user sends $(C, \alpha, \hat{\beta}, \epsilon)$ to the signer. These information allow the verification of the signature with respect to C. The message is still statistically hidden by the hiding property of com because the user never reveals the decommitment r.

Hence, the protocol hides the to-be-signed message and subsequent runs of the protocol for the same message are statistically independent. □

Furthermore, our scheme already supports selective failure blindness as shown in [FS09] because we are signing commitments instead of the adversely chosen messages and even the fourth, additional, move does not reveal any information about the message due to the hiding property of the commitment. The commitment can be safely omitted whenever the messages are just random values without any meaningful content, such as in an e-cash scheme. The user simply starts over with a fresh message.

One-more Unforgeability

In this section, we show that BS is one-more unforgeable, provided that the collision problem $\mathrm{COL}(\mathcal{H}(\mathbf{R}, m), D)$ is hard and the commitment scheme is binding. The main tool in the reduction is the Forking Lemma [PS00, BN06]. It is repeated at the end of this chapter as Lemma 4.12 in Section 4.4.4 for the reader's convenience. To simulate the environment, especially blind signature queries, for the attacker $\mathcal{A}$ in the unforgeability experiment, we require that there are at least two possible secret keys for each public key $\mathbf{S}$ (Lemma 4.6). Moreover, we need the signature protocol to be witness indistinguishable to prevent the attacker from learning the secret key (Lemma 4.7). The binding property of com is necessary to prevent an attacker from obtaining one signature that works for two messages by changing the message under the commitment. All other attackers necessarily output at least one signature that does not correspond to a completed interaction. Here, we apply the Forking Lemma to extract knowledge about the secret key that was used to compute the forgery. Using this knowledge, the reduction can solve the collision problem. Finally, we need to deal with Step 5 in the protocol. The adversary proves that it was unable to obtain a valid signature. We show that this is sufficient if COL is hard.

Since the function family $\mathcal{H}(\mathbf{R}, m)$ compresses the domain D_s^m, it is easy to show that all secret keys collide with at least one other secret key.

Lemma 4.6. *Let $h \in \mathcal{H}(\mathbf{R}, m)$. For every secret key $\hat{\mathbf{s}} \leftarrow_{\$} D_s^m$, there is a second $\hat{\mathbf{s}}' \in D_s^m \setminus \{\hat{\mathbf{s}}\}$ with $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{s}}')$ (with overwhelming probability).*

The next lemma establishes witness indistinguishability of the protocol. Witness indistinguishability ensures that the malicious user cannot distinguish whether the signer uses one of two possible secret keys $\hat{\mathbf{s}}, \hat{\mathbf{s}}' \in h^{-1}(\mathbf{S}) \cap D_s^m$. Basically, it can be interpreted as an application of Lemma 4.4 to $\hat{\mathbf{z}}^* = (\hat{\mathbf{s}}\epsilon^*) + \hat{\mathbf{y}} \in G_*^m$ with some further observations. The choice of $\hat{\mathbf{y}} \leftarrow_{\$} D_y$ and the restriction "$\in G_*^m$" hide the first summand.

Lemma 4.7. *Let $h \in \mathcal{H}(\mathbf{R}, m)$ and $\mathbf{S} \in \mathbf{R}$. For any message msg and any two secret keys $\hat{\mathbf{s}}, \hat{\mathbf{s}}' \in D_s^m$ with $h(\hat{\mathbf{s}}) = \mathbf{S} = h(\hat{\mathbf{s}}')$, the resulting protocol views $(\mathbf{Y}, \epsilon^*, \hat{\mathbf{z}}^*)$ and $(\mathbf{Y}', \epsilon^{*\prime}, \hat{\mathbf{z}}^{*\prime})$ are indistinguishable.*

Using lemmas 4.6 and 4.7, we can exploit witness indistinguishability to simulate all blind signature oracle queries with a secret key $\hat{\mathbf{s}}$ and at the same time expect the adversary to output a forgery that corresponds to a different secret key $\hat{\mathbf{s}}'$ with non-negligible probability or break the binding property of the commitment scheme.

Theorem 4.8 (One-more unforgeability)**.** *Let* OSign *be the signature oracle. Let* T_{OSign} *and* T_{H} *be the cost functions for simulating the oracles* OSign *and* H*, and let* $c < 1$ *be the probability for a restart in the protocol. Let* Q_{H} *be the number of queries to* H*. BS is* $(t, Q_{\mathsf{Sign}}, \delta)$*-one-more unforgeable if* com *is* $(t', \delta/2)$*-binding and* $COL(\mathcal{H}(\mathbf{R}, m), D)$ *is* $(t', \delta'/2)$*-hard with* $t' = t + Q_{\mathsf{H}}^{Q_{\mathsf{OSign}}}(Q_{\mathsf{OSign}}T_{\mathsf{OSign}} + Q_{\mathsf{H}}T_{\mathsf{H}})$ *and non-negligible* δ' *if* δ *is non-negligible.*

The probability δ' depends on the number of issued signatures. It can be found at the end of the proof.

Proof. Towards contradiction, we assume that there exists a successful forger $\mathcal{A}$ against one-more unforgeability of BS with non-negligible probability δ. Using $\mathcal{A}$, we construct an algorithm $\mathcal{B}$, such that it either solves the collision problem or breaks the binding property of com.

Setup: $\mathcal{B}$ flips a coin $b \leftarrow_{\$} \{0,1\}$. For $b = 0$, it selects $h \leftarrow_{\$} \mathcal{H}(\mathbf{R}, m)$. For $b = 1$, it gets the description of h as input. $\mathcal{B}$ initializes a list $L_{\mathsf{H}} \leftarrow \emptyset$ of query-hash pairs $(\mathbf{R} \times \{0,1\}^n, D_\epsilon)$. It chooses $\hat{\mathbf{s}} \leftarrow_{\$} D_s^m$ and sets $\mathbf{S} \leftarrow h(\hat{\mathbf{s}})$. Furthermore, it randomly pre-selects random oracle answers $\mathbf{h}_1, \ldots, \mathbf{h}_{Q_{\mathsf{H}}} \leftarrow_{\$} D_\epsilon$ and a random tape ρ. It runs $\mathcal{A}(\mathbf{S}; \rho)$ in a black-box simulation.

Random Oracle Queries: On input $(\mathbf{u}, C)$, $\mathcal{B}$ looks up $(\mathbf{u}, C)$ in L_{H}. If it finds corresponding hash value ϵ then it returns ϵ. Otherwise, $\mathcal{B}$ selects the first unused ϵ from the list $\mathbf{h}_1, \ldots, \mathbf{h}_{Q_{\mathsf{H}}}$, stores $((\mathbf{u}, C), \epsilon)$ in L_{H}, and returns ϵ.

Blind Signature Queries: $\mathcal{B}$ acts according to the protocol in Figure 4.2.

Output: Eventually, $\mathcal{A}$ stops and outputs $(msg_1, (r_1, \hat{\mathbf{z}}_1, \epsilon_1)), \ldots, (msg_\jmath, (r_\jmath, \hat{\mathbf{z}}_\jmath, \epsilon_\jmath))$, $Q_{\mathsf{OSign}} + 1 = \jmath$, for distinct messages. If $b = 0$, the reduction looks for two pairs $(msg_1^*, (r_1^*, \hat{\mathbf{z}}^*, \epsilon^*))$ and $(msg_2^* \neq msg_1^*, (r_2^*, \hat{\mathbf{z}}^*, \epsilon^*))$ and outputs (msg_1^*, r_1^*), (msg_2^*, r_2^*) to break the binding property of com. If there is no such collision, $\mathcal{B}$ aborts. If $b = 1$, the simulator $\mathcal{B}$ looks for a distinct pair of signatures with the same ϵ. Such a pair directly solves the collision problem. Otherwise, all ϵ_i, $i \in [\jmath]$ are distinct and $\mathcal{B}$ guesses an index $k \leftarrow_{\$} [\jmath]$ such that $h_\imath = \epsilon_k$ for some $\imath \in [Q_{\mathsf{H}}]$. Then, $\mathcal{B}$ starts over, running $\mathcal{A}(\mathbf{S}; \rho)$ with random oracle answers $\mathbf{h}_1, \ldots, \mathbf{h}_{\imath-1}, \mathbf{h}'_\imath, \ldots, \mathbf{h}'_{Q_{\mathsf{H}}}$ for a fresh set $\mathbf{h}'_\imath, \ldots, \mathbf{h}_{Q_{\mathsf{H}}} \leftarrow_{\$} D_\epsilon$. Both $\mathcal{A}$ and $\mathcal{B}$ are run with the same random tape as in the first run. Among other values, $\mathcal{A}$ outputs $(msg'_k, (r'_k, \hat{\mathbf{z}}'_k, \epsilon'_k))$ and $\mathcal{B}$ returns $(\hat{\mathbf{z}}_k - \hat{\mathbf{s}}\epsilon_k, \hat{\mathbf{z}}'_k - \hat{\mathbf{s}}\epsilon'_k)$ if $\hat{\mathbf{z}}_k - \hat{\mathbf{s}}\epsilon_k \neq \hat{\mathbf{z}}'_k - \hat{\mathbf{s}}\epsilon'_k$ in an attempt to solve $\mathrm{COL}(\mathcal{H}(\mathbf{R}, m), D)$. The reduction retries at most $Q_{\mathsf{H}}^\jmath$ times with a different random oracle.

Analysis. $\mathcal{A}$'s environment is perfectly simulated. Especially, restarts happen with the same probability as in the original protocol. For $b = 0$, $\mathcal{B}$ $(t', \delta/2)$-breaks the binding property of com if $\mathcal{A}$ breaks the binding property of com to break one-more unforgeability.

For $b = 1$, we assume that $\mathcal{A}$ breaks one-more unforgeability without attacking com. So, at least one of the output signatures is not obtained via an interaction. The probability that $\mathcal{B}$ guesses the index k of this signature correctly is at least $1/(Q_{\mathsf{OSign}} + 1)$. Observe that ϵ_k is a random oracle answer but with probability $1/|D_\epsilon|$. In total, there are $Q^{\jmath}_{\mathsf{H}}$ index maps $\{(\imath, k) : h_\imath = \epsilon_k\}$. Hence, one of the re-runs of $\mathcal{A}$ yields the same map as in the first run of $\mathcal{A}$ and we can consider the indices in both "interesting" replays to be constant.

Applying the Forking Lemma, we know that with probability $\delta_{\mathsf{frk}} \geq (1-c)(\delta - 1/|D_\epsilon|)((\delta - 1/|D_\epsilon|)/Q_{\mathsf{H}} - 1/|D_\epsilon|)$, $\mathcal{A}$ is again successful in the one-more unforgeability experiment and outputs $(msg'_k, (r'_k, \hat{\mathbf{z}}'_k, \epsilon'_k))$ using the *same random oracle query* as in the first run. The additional $(1-c)$ factor takes a potential abort during the second run into account, which happen with probability at most c. Therefore, we know that $(h(\hat{\mathbf{z}}_k) - \mathbf{S}\epsilon_k, \mathsf{com}(msg_k; r_k)) = (h(\hat{\mathbf{z}}'_k) - \mathbf{S}\epsilon'_k, \mathsf{com}(msg'_k; r'_k))$.

Now, we turn to solving the collision problem. We have to show that $\hat{\mathbf{z}}_k - \hat{\mathbf{s}}\epsilon_k \neq \hat{\mathbf{z}}'_k - \hat{\mathbf{s}}\epsilon'_k$ and $h(\hat{\mathbf{z}}_k - \hat{\mathbf{s}}\epsilon_k) = h(\hat{\mathbf{z}}'_k - \hat{\mathbf{s}}\epsilon'_k)$. The second requirement follows directly from the previous paragraph. The first is more involved. Here, it is important that the protocol is witness indistinguishable (Lemma 4.7), i.e., the adversary does not recognize whether we have used one of at least two possible $\hat{\mathbf{s}}, \hat{\mathbf{s}}'$ (Lemma 4.6 with probability greater than $1/2$. Thus, with probability at least $1/2$ its output corresponds to $\hat{\mathbf{s}}'$. More precisely, we need that, for at least one $\hat{\mathbf{t}} \in \{\hat{\mathbf{s}}, \hat{\mathbf{s}}'\}$, the random variables $\chi_k = \hat{\mathbf{z}}_k - \hat{\mathbf{t}}\epsilon_k$ and $\chi'_k = \hat{\mathbf{z}}'_k - \hat{\mathbf{t}}\epsilon'_k$ are sensitive to the modified random oracle answers for indices $\geq \imath$. Hence, $\chi_k \neq \chi'_k$ with probability at least $1/2$ and we obtain the desired collision with norm at most $d_G + nd_sd_\epsilon < d_D$. Otherwise, we would have $\hat{\mathbf{z}}_k - \hat{\mathbf{s}}\epsilon_k = \hat{\mathbf{z}}'_k - \hat{\mathbf{s}}\epsilon'_k$ and $\hat{\mathbf{z}}_k - \hat{\mathbf{s}}'\epsilon_k = \hat{\mathbf{z}}'_k - \hat{\mathbf{s}}'\epsilon'_k$. We subtract the equations and obtain $(\epsilon_k - \epsilon'_k)(\hat{\mathbf{s}}' - \hat{\mathbf{s}}) = \mathbf{0}$. We know that $\epsilon_k - \epsilon'_k \neq \mathbf{0}$. Now, $\|(\epsilon_k - \epsilon'_k)(\hat{\mathbf{s}}' - \hat{\mathbf{s}})\|_\infty \leq 4d_sn < q/2$ because $\|\epsilon_k - \epsilon'_k\|_\infty \leq 2$ and $\|\hat{\mathbf{s}}' - \hat{\mathbf{s}}\|_\infty \leq 2d_s$. Thus, $(\epsilon_k - \epsilon'_k)(\hat{\mathbf{s}}' - \hat{\mathbf{s}}) = \mathbf{0}$ over $\mathbb{Z}[X]/\langle X^n + 1\rangle$, which is an integral domain. So, we have the contradiction $\hat{\mathbf{s}}' = \hat{\mathbf{s}}$ and a collision $(\hat{\mathbf{z}}_k - \hat{\mathbf{s}}\epsilon_k, \hat{\mathbf{z}}'_k - \hat{\mathbf{s}}\epsilon'_k) \in D \times D$. The success probability is at least $\delta_{\mathsf{col}} \geq 1/2\,\delta_{\mathsf{frk}}/(Q_{\mathsf{OSign}} + 1)$, which is non-negligible if δ is non-negligible.

Concerning restarts, we argue that the user cannot obtain a valid signature out of an aborted interaction without solving the collision problem. In order to trigger an abort after Step 4, it outputs $result = (C, \alpha, \hat{\beta}, \epsilon)$ which, together with $\hat{\mathbf{z}}^*, \hat{\mathbf{y}}, \epsilon^*$,

satisfies all abort criteria:

$$\epsilon^* + \alpha = \epsilon \quad = \quad \mathsf{H}(\mathbf{Y} - \mathbf{S}\alpha - h(\hat{\beta}), C) \tag{4.1}$$
$$\epsilon \quad = \quad \mathsf{H}(h(\hat{\mathbf{z}}^* - \hat{\beta}) - \mathbf{S}\epsilon, C) \tag{4.2}$$
$$\hat{\mathbf{z}}^* - \hat{\beta} \quad \notin \quad G^m \tag{4.3}$$

Assume that it also obtains a valid signature $(r', \hat{\mathbf{z}}', \epsilon')$ from this interaction. If $\epsilon = \epsilon'$, then $h(\hat{\mathbf{z}}^* - \hat{\beta} - \hat{\mathbf{s}}\epsilon) = h(\hat{\mathbf{z}}' - \hat{\mathbf{s}}\epsilon)$ by (4.2). If the arguments under h are equal, we have $\hat{\mathbf{z}}^* - \hat{\beta} \in G^m$ — a contradiction with (4.3). If the arguments are distinct, we have a collision in D because $\|\hat{\mathbf{z}}' - \hat{\mathbf{s}}\epsilon\|_\infty \leq d_G < d_D$ and $\left\|\hat{\mathbf{z}}^* - \hat{\beta}^* - \hat{\mathbf{s}}\epsilon\right\|_\infty \leq d_{G_*} + d_\beta + n d_s d_\epsilon = d_D$.

The adversary may succeed by hiding $\epsilon' \neq \epsilon$ in ϵ^*. But then, we necessarily have $\epsilon^* = \epsilon - \alpha = \epsilon' - \alpha'$ by (4.1) for an $\alpha \neq \alpha'$ and we know that $\alpha = \epsilon - \epsilon' + \alpha'$. So, the adversary had to be able to predict the output of H to compute α.

To conclude, the probability that we can extract a collision from a cheating user during an abort is at least $\delta_{\mathsf{abort}} \geq \delta\,(1 - 1/|D_\epsilon|)$, which is non-negligible if δ is non-negligible. Thus, the overall success probability of the reduction is $\delta' \geq \min(\delta_{\mathsf{col}}, \delta_{\mathsf{abort}})$ if the guess $b = 1$ was correct. □

Hence, we require that $Q_{\mathsf{OSign}} = o(n)$ to be able to rely on the subexponential hardness of lattice problems. This constraint is an artifact of the proof technique as discussed in [PS00] and it is not at all unusual for efficient blind signature schemes. There, it was even required that $Q_{\mathsf{OSign}} \leq (\log(n))^{\mathcal{O}(1)}$ because they needed a polynomial-time reduction. In consequence, in our reduction, we greatly benefit from the subexponential hardness of the underlying lattice problem. Alternatively, we believe that the running time of the reduction can be significantly reduced to being polynomial in q_{OSign} by using techniques due to Pointcheval [Poi98]. We leave this as an open research question.

Via Proposition 2.3, we get the following strong worst-case security guarantees.

Corollary 4.9. *BS is one-more unforgeable if solving SIVP$^\infty$ is hard in the worst case for approximation factors $\gamma = \widetilde{\mathcal{O}}(n^5)$ in lattices that correspond to ideals in $\mathbf{R}_0$.*

Leakage Resilience

Using an additional restriction for one of the parameters, we can safely leak a $(1 - o(1))$ fraction of the secret key in the unforgeability experiment according to the definition in Section 4.1.1. The following results also carry over to Lyubashevsky's

identification and signature schemes [Lyu09]. Recall that $m = \lfloor c_m \log(q) \rfloor + 1$ for some $c_m = \widetilde{\mathcal{O}}(1)$. Thus, it is possible to choose c_m, say $\log(n)$, without loosing the scheme's quasi-optimal efficiency. The following theorem states that such a choice is sufficient to provide strong leakage resilience.

To prove the theorem, we use a technical lemma from [KV09] in its alternative interpretation.

Lemma 4.10 ([KV09, Lemma 1])**.** *Let X be a random variable with $H := H_\infty(X)$, and fix $H' \in [0, H]$. Let f be a function whose range has size 2^λ, and set $Y := \{y \in \{0,1\}^\lambda | H_\infty(X|y = f(X)) \geq H'\}$. Then* $\mathrm{Prob}[\, f(X) \in Y] \geq 1 - 2^{\lambda - H + H'}$.

In our context, it states that the conditional min-entropy of the secret key after λ bits of leakage is at least H' but with probability $2^{\lambda - H + H'}$. We will use the lemma with $H' = 1$ because one bit of uncertainty is sufficient to apply Lemma 4.7 (witness indistinguishability) in Theorem 4.8.

Theorem 4.11 (Leakage Resilience)**.** *Let $c_m = \omega(1)$ and let $L := \log(|D_s^m|) = mn \log(2d_s + 1)$ be the length of the secret key. The conditional min-entropy H_∞ of $\hat{\mathbf{s}}$, conditioned on $\mathbf{S} = h(\hat{\mathbf{s}})$ and a total secret-key leakage $f(\hat{\mathbf{s}})$ of $\lambda = \delta L = (1 - o(1))L$ bits, is positive with overwhelming probability.*

Proof. We prove a conservative lower bound on the amount of tolerated leakage because we treat the public key $\mathbf{S}$ as additional leakage. Therefore, we define a new total leakage function $f'(\hat{\mathbf{s}}) = f(\hat{\mathbf{s}}) \| h(\hat{\mathbf{s}})$ with a total leakage of at most $\lambda' = \lambda + n \log(q)$ bits. Now, we apply Lemma 4.10 to f', λ', and H' with $\hat{\mathbf{s}}$ being the random variable. Observe that $H = L = mn \log(2d_s + 1)$. It yields

$$\mathrm{Prob}\left[\, f'(\hat{\mathbf{s}}) \in Y\right] \geq 1 - 2^{\lambda + n \log(q) - L + 1}\,, \tag{4.4}$$

which we want to be overwhelming $\geq 1 - 2^{-p(n)}$. We take any function $p(n)$, $\omega(\log(n)) \leq p(n) \leq \mathcal{O}(n \log(n))$ and bound the relative leakage $\delta \leq 1 - \frac{p(n) + n \log(q) + 1}{L}$ $= 1 - \frac{\Theta(n \log(n))}{c_m \Theta(n \log(n))} = 1 - \frac{1}{\omega(1)} = 1 - o(1)$.

In consequence, (4.4) becomes $\geq 1 - 2^{\left(1 - \frac{p(n) + n \log(q) + 1}{L}\right) L + n \log(q) - L + 1} = 1 - 2^{p(n)}$. Thus, $\delta L = (1 - o(1))L$ leakage bits yield a non-zero conditional min-entropy with probability $1 - 2^{-p(n)} \geq 1 - 2^{-\omega(\log(n))}$. □

4.3 Practical Parameters

Although worst-case guarantees are a good argument for lattice-based cryptography in general, we need to analyze the underlying average-case problem, the collision

problem, directly before proposing concrete parameters. To this end, we use our results from Chapter 3.

All we need to apply the framework, is to feed it with valid parameter relations (cf. Table 4.2) and run the main reduction (one-more unforgeability). As a result, we end up with an instance of the collision problem $\mathrm{COL}(\mathcal{H}(\mathbf{R}, m), D)$, or alternatively with the SIS (short integer solution) problem. Since Chapter 3 only deals with ℓ_2-norm we relax the adversaries task and assume it only needs to find a vector of Euclidean norm $\leq d_D\sqrt{mn}$. The resulting instance of SIS then yields a hardness estimate δ, which can be related to "bit security".

4.3.1 Optimization

Before we propose actual parameters, we optimize the choice of parameters based on the constraints in Table 4.2. We discuss the parameters q, ψ, ϕ, and d_s.

Choosing q From Table 4.2 and Proposition 2.3, we know that the worst-case to average-case reduction relies on having $q \geq 4mn\sqrt{n}\log(n)d_D = \widetilde{\Theta}(n^5\sqrt{n})$. For practical parameters, we typically arrive at $q \approx n^8$ because of the hidden constants.

Choosing ψ As noted before, we can safely set $\psi = 1$ and let the user handle an expected number of e restarts after Step 2 in the protocol. These restarts are performed locally and experiments show that they do not significantly affect the overall efficiency of the protocol.

Choosing ϕ Here, the situation is different because it controls the probability of a restart after Steps 3 and 4. Restarts at this point cannot be done locally and involve an increase in communication and computation costs. The influence becomes obvious when looking at Figure 4.3. It shows the number of required repetitions of the blind signature protocol for $n \in \{1024, 2048, 4096\}$ and $1 \leq \phi \leq 15$, averaged over 1000 random instances. As a comparison, it also shows the estimated number of repetitions $e^{2/\phi}$ from Section 4.2. As expected, the behavior is almost independent of n. Our experiments show that one should choose $\phi \geq 4$ and by closely looking at the numbers it even makes sense to choose $\phi = 10$ in some cases.

This observation is backed up by Figure 4.4, which shows the actual combined running time in milliseconds of signing and verification, averaged over 1000 random instances for each pair (n, ϕ). Since our implementation is pretty straightforward without many optimizations, we believe that there is a lot of room for improvements.

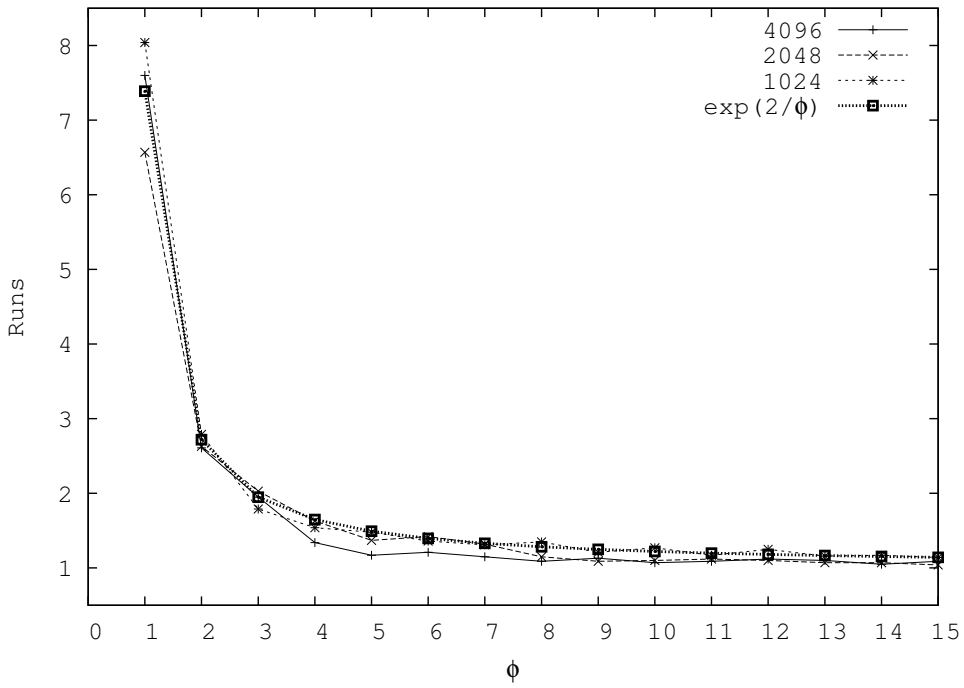


Figure 4.3: Average number of runs needed to complete in the signature protocol of BS for $1 \leq \phi \leq 15$ and $n \in \{1024, 2048, 4096\}$.

When increasing ϕ, one has to keep in mind that a larger ϕ also increases d_D and requires a slightly stronger hardness assumption.

Choosing d_s When looking at the constraint for m, basically $m > \log(q)/\log(2d_s)$, it is clear that a larger d_s allows us to choose a smaller m. This can dramatically decrease the required bandwidth and computational cost at the expense of a slightly larger secret key. We show the effect of a larger d_s in Figure 4.5 for $n = 2048$ and $\phi = 4$. Notice that we take the logarithm of d_s and round m to the next integer. So, in order to decrease m by a factor k, we need to choose d_s around 2^k. When removing the outliers, this leads to a perfect stair-stepped graph, where the steps become wider with increasing d_s. Also, this increase comes at the price of having a larger d_D and requires stronger hardness assumption.

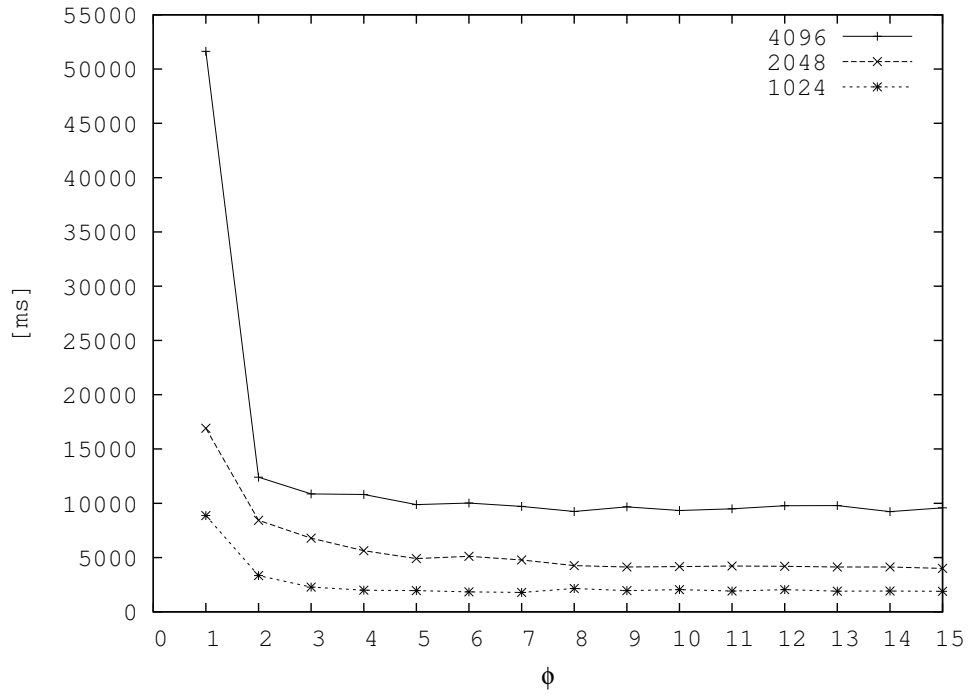


Figure 4.4: Average running time of signature protocol and the verification algorithm of BS for $1 \leq \phi \leq 15$ and $n \in \{1024, 2048, 4096\}$ in milliseconds, including eventual restarts.

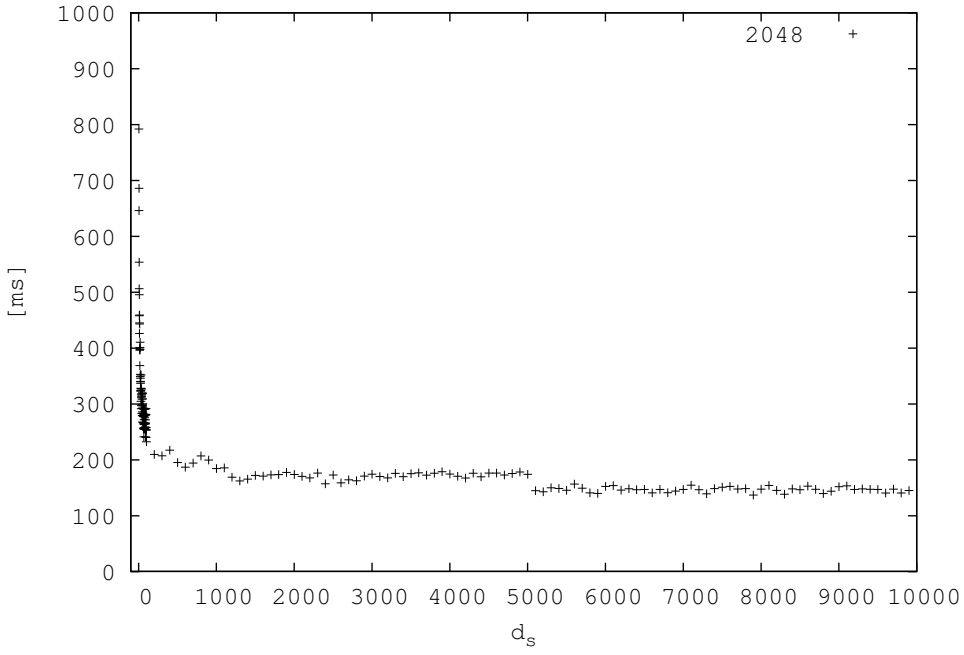


Figure 4.5: Average running time of signature protocol and the verification algorithm of BS for $1 \leq d_s \leq 10{,}000$ and $n = 2048$ and $\phi = 4$. The range is truncated to $[0, 1000]$.

Parameter	Current I	Current II	Current III	Medium I	Medium II	Medium III
n	1024	1024	1024	2048	2048	2048
q	$\approx 2^{78}$	$\approx 2^{85}$	$\approx 2^{81}$	$\approx 2^{85}$	$\approx 2^{91}$	$\approx 2^{94}$
ϕ	1	8	4	1	10	4
d_s	1	1	283	1	1	241080
m	79	86	9	85	92	5
Repetitions	7.13	*1.32*	1.55	7.67	*1.16*	1.23
Secret key	15.7 kB	17 kB	10.3 kB	33.7 kB	36.5 kB	23.6 kB
Public key	9.8 kB	10.6 kB	10,2 kB	21.2 kB	22.9 kB	23.6 kB
Signature	529.4 kB	643.4 kB	*66.9 kB*	1228.6 kB	1487.9 kB	*89.4 kB*
Communication	2706.4 kB	589.5 kB	*95.2 kB*	6771.5 kB	1199.8 kB	*119.1 kB*
KeyGen	296 ms	369 ms	*37 ms*	674 ms	843 ms	*52 ms*
Signing	7629 ms	1819 ms	*220 ms*	19000 ms	3656 ms	*283 ms*
Verification	226 ms	293 ms	*33 ms*	532 ms	679 ms	*57 ms*

Table 4.3: Exemplary parameters for the blind signature scheme in Section 4.2.

4.3.2 Secure Parameters

Table 4.3 shows a few exemplary parameter sets for current (76 bit) and medium (102 bit) security levels. According to `www.keylength.com`, they correspond to security until the years 2012 and, 2050, respectively. We leave out the detailed parameters for future (256 bit / year 2282) security.

For both security levels, we propose three parameter sets. The first requires the mildest hardness assumption and uses the smallest modulus. The second minimizes the number of repetitions and the third is simultaneously optimized for computational cost and bandwidth including restarts. The optimization goal is denoted in italics. Depending on the application scenario, other trade-offs are possible.

4.4 Supporting Lemmas

In the following, we prove a number of supporting lemmas for our main theorem in Section 4.2.

4.4.1 Completeness

Proving Lemmas 4.1 and 4.2 concludes the discussion of completeness in Theorem 4.3. Lemma 4.1 shows that the number of aborts/restarts in the protocol is small. Lemma 4.2 ensures that multiplying two “short” polynomials only slightly increases

the norm of the resulting coefficient vector.

Lemma 4.1 Let $k = \Omega(n)$, $A, B \in \mathbb{N}_{>0}$, $\mathbf{a}, \mathbf{b} \in \mathbb{Z}^k$ with *arbitrary* $\mathbf{a} \in \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq A\}$ and *random* $\mathbf{b} \leftarrow_\$ \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq B\}$. Given $B \geq \phi k A$ for $\phi \in \mathbb{N}_{>0}$, we have $\operatorname{Prob}_{\mathbf{b}}[\,\|\mathbf{a} - \mathbf{b}\|_\infty \leq B - A] > e^{-1/\phi} - o(1)$.

Proof. Observe that $\operatorname{Prob}[\,\|\mathbf{a} - \mathbf{b}\|_\infty \leq B - A] = \operatorname{Prob}[\,|a_i - b_i| \leq B - A]^k$ and that the b_i need to be in the range $[-(B-A)+a_i, B-A+a_i] \subseteq [-B, B]$ for that. Therefore, the probability is

$$\left(\frac{2(B-A)+1}{2B+1}\right)^k > \left(1 - \frac{A}{B}\right)^k \geq \left(1 - \frac{1/\phi}{k}\right)^k.$$

By the series expansion at infinity, this is at least $\frac{1}{e^{1/\phi}} - o(1)$ for an $o(1)$ term that vanishes like $1/k$. □

Lemma 4.2 For any two $\mathbf{a}, \mathbf{b} \in \mathbb{Z}[X]$ of degree n, we have $\|\mathbf{ab} \bmod (X^n + 1)\|_\infty \leq n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$.

Proof. Note that $\mathbf{c} = \mathbf{ab} \bmod (X^n + 1) = \sum_{i=0}^{n-1} a_i \mathbf{b} X^i \bmod (X^n + 1)$. Hence, we have $\|\mathbf{c}\|_\infty \leq n \|\mathbf{a}\|_\infty \max_{i=0,\ldots,n-1}\{\left\|\mathbf{b}X^i \bmod (X^n + 1)\right\|_\infty\}$. For our particular relation $X^n = -1$, this is easy to evaluate because $\mathbf{b}X = b_0X + b_1X^2 + \cdots + b_{n-2}X^{n-1} + b_{n-1}X^n \bmod (X^n + 1) = -b_{n-1} + b_0X + b_1X^2 + \cdots + b_{n-2}X^{n-1}$. Therefore, we have $\left\|\mathbf{b}X^i \bmod (X^n + 1)\right\|_\infty = \|\mathbf{b}\|_\infty$ and $\|\mathbf{c}\|_\infty \leq n \|\mathbf{a}\|_\infty \|\mathbf{b}\|_\infty$. □

4.4.2 Blindness

Lemma 4.4 establishes blindness in Theorem 4.5. It guarantees that the every output by the user is distributed independently of the signed message.

Lemma 4.4 Let $k \in \mathbb{N}_{>0}$, $A, B \in \mathbb{N}_{>0}$, $\mathbf{a}, \mathbf{a}', \mathbf{b} \in \mathbb{Z}^k$ with *arbitrary* $\mathbf{a}, \mathbf{a}' \in \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq A\}$, a *random* $\mathbf{b} \leftarrow_\$ \{\mathbf{v} \in \mathbb{Z}^k : \|\mathbf{v}\|_\infty \leq B\}$ for $B > A$. We define the random variables $\mathbf{c} \leftarrow \mathbf{a} - \mathbf{b}$ and $\mathbf{c}' \leftarrow \mathbf{a}' - \mathbf{b}$ if $\max\{\|\mathbf{a} - \mathbf{b}\|_\infty, \|\mathbf{a}' - \mathbf{b}\|_\infty\} \leq B - A$, otherwise, we resample $\mathbf{b}$. Then, $\Delta(\mathbf{c}, \mathbf{c}') = 0$.

Proof. By definition, the statistical distance is

$$\begin{aligned} & \frac{1}{2} \sum_{\mathbf{c}:\|\mathbf{c}\|_\infty \leq B-A} \left| \operatorname*{Prob}_{\mathbf{b}}[\mathbf{a}-\mathbf{b}=\mathbf{c}] - \operatorname*{Prob}_{\mathbf{b}}\left[\mathbf{a}'-\mathbf{b}=\mathbf{c}\right] \right| \\ = \quad & \frac{1}{2} \sum_{\mathbf{c}} \left| \operatorname*{Prob}_{\mathbf{b}}[\mathbf{b}=\mathbf{a}-\mathbf{c}] - \operatorname*{Prob}_{\mathbf{b}}\left[\mathbf{b}=\mathbf{a}'-\mathbf{c}\right] \right| . \end{aligned}$$

Observe that $\max\{\|\mathbf{a}-\mathbf{c}\|_\infty, \|\mathbf{a}'-\mathbf{c}\|_\infty\} \leq A+(B-A) = B$ and $\|\mathbf{b}\|_\infty \leq B$. Hence, the probability in either case is $1/(2B+1)^k$ and the statistical distance is 0. □

4.4.3 One-more Unforgeability

Lemma 4.6 ensures that all secret keys collide with at least one alternative secret key under h. In combination with Lemma 4.7, which proves witness indistinguishability of the signature issue protocol, this allows us to build a reduction algorithm in Theorem 4.8 that correctly simulates the signer and breaks $\mathrm{COL}(\mathcal{H}(\mathbf{R}, m), D)$ with the help of a forger.

Lemma 4.6 Let $h \in \mathcal{H}(\mathbf{R}, m)$. For every secret key $\hat{\mathbf{s}} \leftarrow_\$ D_s^m$, there is a second $\hat{\mathbf{s}}' \in D_s^m \setminus \{\hat{\mathbf{s}}\}$ with $h(\hat{\mathbf{s}}) = h(\hat{\mathbf{s}}')$ (with overwhelming probability).

Proof. All functions in the family $\mathcal{H}(\mathbf{R}, m)$ are compressing when applied to the domain D_s^m for our choice of parameters because $|D_s^m| = (2d_s+1)^{mn} > 3^{mn} > (2d_s)^{n\log(q)/\log(2d_s)} > 2^{n\log(q)} = q^n = |\mathbf{R}|$. Therefore, all but at most q^n elements in D_s^m do not collide. The probability of selecting such an element is at most $(q/(2d_s+1)^m)^n < 2^{-n\log(q)\log(2d_s+1)/\log(2d_s)} < 2^{-n\log(q)} = 2^{-\Omega(n\log(n))}$. □

Lemma 4.7 Let $h \in \mathcal{H}(\mathbf{R}, m)$ and $\mathbf{S} \in \mathbf{R}$. For any message msg and any two secret keys $\hat{\mathbf{s}}, \hat{\mathbf{s}}' \in D_s^m$ with $h(\hat{\mathbf{s}}) = \mathbf{S} = h(\hat{\mathbf{s}}')$, the resulting protocol views $(\mathbf{Y}, \epsilon^*, \hat{\mathbf{z}}^*)$ and $(\mathbf{Y}', \epsilon^{*\prime}, \hat{\mathbf{z}}^{*\prime})$ are indistinguishable.

Proof. The argument is an adaptation of [Lyu08b, Theorem 5.5]. We interpret the components of the view as random variables. Firstly, observe that $\mathbf{Y}$ and $\mathbf{Y}'$ are chosen independently of the secret key. Secondly, ϵ^* and $\epsilon^{*\prime}$ are independent of a particular $\hat{\mathbf{y}} \in h^{-1}(\mathbf{Y}) \cap D_y^m$ because $\mathbf{Y}$ statistically hides $\hat{\mathbf{y}}$. Finally, we need to argue about the indistinguishability of $\hat{\mathbf{z}}^*$ and $\hat{\mathbf{z}}^{*\prime}$. Let ϵ^* be any challenge and let $\hat{\mathbf{z}}^* = \hat{\mathbf{s}}\epsilon^* + \hat{\mathbf{y}} \in G_*^m$. Then, we can set $\hat{\mathbf{y}}' \leftarrow \hat{\mathbf{y}} + \hat{\mathbf{s}}\epsilon^* - \hat{\mathbf{s}}'\epsilon^*$ for which $\hat{\mathbf{z}}^* = \hat{\mathbf{s}}'\epsilon^* + \hat{\mathbf{y}}'$. We need to show that $\hat{\mathbf{y}}' \in h^{-1}(\mathbf{Y}) \cap D_y^m$. This implies that for every $\hat{\mathbf{y}}$ (for $\hat{\mathbf{s}}$),

there is a $\hat{\mathbf{y}}'$ (for $\hat{\mathbf{s}}'$) that yields the same output. Thus, the probability of a restart would also be equal. Clearly, $\hat{\mathbf{y}}' \in h^{-1}(\mathbf{Y})$ because $h(\hat{\mathbf{y}}') = \mathbf{Y} + \mathbf{S}\epsilon^* - \mathbf{S}\epsilon^* = \mathbf{Y}$. Furthermore, $\|\hat{\mathbf{y}}'\|_\infty \leq \|\hat{\mathbf{z}}^*\|_\infty + \|\hat{\mathbf{s}}'\epsilon^*\|_\infty \leq d_y - nd_sd_{\epsilon^*} + nd_sd_{\epsilon^*} = d_y$ by Lemma 4.2 and we can conclude $\hat{\mathbf{y}}' \in D_y^m$. □

4.4.4 Forking Lemma

The generalized Forking Lemma of Bellare and Neven [BN06] is a tool for proving security in the random oracle model. It provides a lower bound for the probability that a randomized algorithm outputs two related values when run twice with the same random tape but with a different random oracle.

Lemma 4.12 (Lemma 1 in [BN06])**.** *Fix an integer $Q \geq 1$ and a set H of size $h \geq 2$. Let $\mathcal{A}$ be a randomized algorithm that on input $x, h_1, \ldots, h_Q$ returns a pair, the first element of which is an integer in the range $0, \ldots, Q$ and the second element of which we refer to as a side output. Let IG be a randomized algorithm that we call the input generator. The accepting probability of $\mathcal{A}$, denoted acc, is defined as the probability that $J \geq 1$ in the experiment $x \leftarrow_\$ \mathsf{IG}; h_1, \ldots, h_Q \leftarrow_\$ H; (J, \sigma) \leftarrow_\$ \mathcal{A}(x, h_1, \ldots, h_Q)$. The forking algorithm $\mathsf{F}_\mathcal{A}$ associated to $\mathcal{A}$ is the randomized algorithm that takes input x proceeds as follows:*

Algorithm $\mathsf{F}_\mathcal{A}(x)$
Pick coins ρ for $\mathcal{A}$ at random
$h_1, \ldots, h_Q \leftarrow_\$ H^Q$
$(I, \sigma) \leftarrow \mathcal{A}(x, h_1, \ldots, h_Q; \rho)$
If $I = 0$ then return $(0, \epsilon, \epsilon)$
$h'_I, \ldots, h'_Q \leftarrow_\$ H$
$(I', \sigma') \leftarrow \mathcal{A}(x, h_1, \ldots, h_{I-1}, h'_I, \ldots, h'_Q; \rho)$
If $I = I'$ and $h_I \neq h'_{I'}$ then return $(1, \sigma, \sigma')$
Else return $(0, \epsilon, \epsilon)$.

Let $\mathsf{frk} = \mathrm{Prob}[\, b = 1 : x \leftarrow_\$ \mathsf{IG}; (b, \sigma, \sigma') \leftarrow \mathsf{F}_\mathcal{A}(x)]$. *Then* $\mathsf{frk} \geq \mathsf{acc}\left(\frac{\mathsf{acc}}{Q} - \frac{1}{h}\right)$.

4.5 Conclusion and Open Problems

We have shown how to construct an efficient and provably secure blind signature scheme based on the hardness of worst-case lattice problems. Our scheme has four moves, offers quasi-optimal performance, and it is leakage resilient in an almost optimal sense.

Further research directions include the elimination of the random oracle, further efficiency improvements, and similar constructions of fair blind [SPC95] and partially blind [AF96] signature schemes. A standard model blind signature scheme from lattices is likely to involve fundamentally new techniques. Modifications to fair or partially blind signatures seem within reach. Here, it would be interesting to try and instantiate our unified model in [RS10b] for fair partially blind signatures. The dissertation author was the primary investigator and author of this paper. In this model, the signer can control parts of the signed message, e.g., an expiry date for a blindly issued voucher. At the same time, if a user misbehaves or even commits a crime with a blind signature, a trusted authority can trace the corresponding interaction back to the fraudulent user.

Chapter 5

Verifiably Encrypted Signatures

Boneh et al. introduce the concept of verifiably encrypted signatures (VES) as a means of covertly exchanging signatures, while maintaining their verifiability [BGLS03]. They include a passive, trusted third party, the adjudicator, which makes VES schemes fall into the category of optimistic fair exchange protocols [ASW00, BDM98].

Signer, receiver, and adjudicator, interact as follows. The signer encrypts his or her signature σ for a document *msg* as a verifiably encrypted signature ϖ. Given ϖ, the receiver can verify that it contains a valid signature for *msg*, but is otherwise unable to extract σ. A commercially important application is *online contract signing* under the supervision of an escrow. As opposed to the classic (offline) scenario, where the escrow needs to be involved in every step of the process, verifiably encrypted signatures make signing contracts more cost-efficient due to the passiveness of the escrow and simultaneously allow the parties to negotiate online rather than meet in person. They simply exchange verifiably encrypted signatures on the negotiated contract, verify them, and finally exchange the corresponding regular signatures. Assume Alice acts honestly, while Bob tries to misuse (e.g., for negotiating a better deal elsewhere or simply for blackmail) the partially signed document without providing a signature himself. The adjudicator can step in and disclose Bob's signature. The contract becomes binding despite Bob's attempt to back out.

Boneh et al. also propose the first construction, which is provably secure in the random oracle model, followed by a slightly more efficient construction by Zhang et al. [ZSNS03]. Both use pairings. Lu et al. present the first scheme in the standard model in [LOS$^+$06]. Furthermore, they sketch a generic construction based on non-interactive zero-knowledge (NIZK) proofs. Another NIZK construction has been proposed by Dodis et al. in [DLY07]. Using NIZKs, however, is generally very inefficient with respect to computational cost and signature size.

In [BGLS03], security of is defined via unforgeability and opacity. Roughly speaking, unforgeability assures that a malicious user cannot produce signatures on behalf of another party. Opacity guarantees that only the adjudicator and the signer can disclose a signature from a verifiably encrypted signature.

Surprisingly, the original security model does *not* guarantee that the adjudicator is always able to extract a valid signature from a valid verifiably encrypted signature. In fact, we show that every VES can easily be turned into a scheme which remains secure in this model, but where a malicious signer can output a verifiably encrypted signature such that the ordinary signature is hidden irrecoverably — a disaster for optimistic fair exchange.

Our Contribution The results in this chapter are essentially bifold. We improve the initial, flawed security model for VES schemes and we propose a generic, modular framework for constructing VES schemes. In particular, we show that there is a lattice-based instantiation in the standard model.

New Security Model As our first result, we extend the model of [BGLS03] to ensure the aforementioned *extractability*. In addition, we propose *non-frameability* as a stronger form of unforgeability. Basically, a non-frameable VES scheme guarantees that an adversary who cooperates with the adjudicator is not able to derive a verifiably encrypted signature on behalf of an honest signer. We show that non-frameability is indeed stronger than unforgeability and that for a "natural" class of *extractable* VES schemes, non-frameability is already implied. Since the instantiation of [BGLS03] and [LOS+06] fall into this class, our results also give more confidence about the security of their schemes.

The second part of the chapter is devoted to generic constructions. A common approach would be: take a message, sign it, encrypt the signature, and append a NIZK, proving that decryption yields a valid signature for the given message. However, we are interested in a standard model construction and, hence, need to avoid NIZK proofs. This entails the following model extension.

VES with a Setup Phase We extend our specification and security model in the sense that the signer's key may depend on the adjudicator's public key and to limit the number of issuable verifiably encrypted signatures. More precisely, we allow signer and adjudicator to interact *once* during signing key generation. We believe that this is a good model of real-world business scenarios. To illustrate this, let us consider a notary, the adjudicator, that oversees fair online contract signing. In general, the notary wants to remain passive but he or she still wants to bill his or her services on a per signature-exchange basis. With our extension and the instantiations therein, we show how the (offline) notary may actually control the number of verifiably encrypted signature that his or her customers can securely exchange. The customer pays for a certain number of signatures in advance and the notary completes the customer's key pair accordingly. Interestingly, the underlying signature scheme can still be used in a black-box way. Thus, smart cards or other signing devices can be used and the secret signing key is not revealed. This is important for contract signing as laws and ordinances often require this for the signed contract to be legally binding.

Generic Construction So far, there have been two construction principles for VES schemes: use pairings for efficiency or NIZKs for generic constructions from minimal cryptographic assumptions. Our construction fills the gap between those extremes as it can be considered both efficient (compared to NIZKs) and generic.

In detail, we show that generic constructions for VES schemes need not involve inefficient non-interactive zero-knowledge proofs. We propose two generic constructions in the standard model. Both are based on "random plaintext secure" (RPA) encryption, "maskable" existentially unforgeable signatures, and a collision-resistant hash function in a Merkle authentication tree [Mer89]. The latter implies a polynomial bound on the number of issuable signatures. On the positive side, it allows us to scale the resulting scheme according to the individual needs of specific application scenarios.

We introduce RPA security as a weaker form of CPA security (cf. Section 2.3.5), i.e., every CPA scheme is also RPA but not vice-versa.

Maskability is a property of a signature scheme, which states that one can choose a random masking value α and mask a given signature σ for a message *msg* by calling $(\tau, \beta) \leftarrow \mathsf{Mask}(\sigma, \alpha)$. The resulting masked signature τ is still valid for the same message under a modified verification procedure that uses the auxiliary advice β. Given (τ, β), it is hard to recover σ. However, with the secret masking value α, one can call $\sigma' \leftarrow \mathsf{Unmask}(\tau, \beta, \alpha)$ and recover a valid (ordinary) signature for *msg*.

Our first construction uses regular (many-time) signature schemes, while the second only requires the existence of a suitable one-time signature scheme. We summarize our result in the following theorem.

Theorem (Generic Construction)**.** *Opaque, extractable, and non-frameable VES schemes with a setup exist if maskable (one-time) signature schemes, RPA secure encryption schemes, and collision-resistant hash functions exist.*

Both constructions are stateful and the key generation step depends, as always with tree-based constructions [Mer89], on the desired signature capacity ℓ of the resulting scheme. Using Merkle trees for VES was first considered in [Rüc09] for the special case of RSA signatures. By formalizing this approach, we develop the technique to its full potential.

A potential instantiation of our generic framework is entirely built upon lattice-based primitives, as stated in the following summarizing theorem.

Theorem (Instantiation)**.** *Let* $D \subseteq \mathbf{R}_0$ *such that* $\mathbf{f} \in D$ *if* $\|\mathbf{f}\|_\infty = \widetilde{\mathcal{O}}(n^2)$*. Our generic construction can be instantiated with ideal lattices in the standard model, provided that:*

- *$COL(\mathcal{H}(\mathbf{R}, m), D)$ is hard;*
- *RPA encryption exists in ideal lattices (e.g., [LPR10]);*
- *Collision-resistant hash functions exist in ideal lattices (e.g., [ADL+08]).*

Organization After discussing our extended specification in Section 5.1, we propose a new, strengthened security model in Section 5.2. Here, we also formally justify the need for the new requirements, extractability and non-frameability, by separating our model from the Boneh et al. model. We conclude the discussion of the security model with two implications within the security model, which simplify the proofs later on. Section 5.3 contains our generic constructions and a discussion of the required ingredients. Then, we propose an exemplary instantiation in Section 5.4 and conclude the chapter in Section 5.5.

This chapter is a partial reprint of [RS09], [Rüc09], and [RSS10]. The dissertation author was the primary investigator and author of these papers. The chapter also contains various improvements to [RSS10], making the proofs more modular. Especially the security properties of maskable signatures are more precise now.

5.1 Definition

Following the original definition due to Boneh et al. [BGLS03] and our extensions [RS09, RSS10], we define a VES as a tuple of algorithms $(\mathsf{AdjKg}, \mathsf{AdjSetup}, \mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf}, \mathsf{Adj})$. Furthermore, a VES scheme builds upon a signature scheme $\mathrm{DS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$.

We generalize the original model [BGLS03] slightly by allowing the key generation algorithm Kg of the signer to depend on the keys of the adjudicator. This *optional* dependency is modeled via an interaction $\mathsf{AdjSetup}$ between signer and adjudicator during Kg. It can be viewed as a *one-time* registration of a signing party with its notary. The specification is as follows.

Adjudicator Key Generation: $\mathsf{AdjKg}(1^n)$ outputs a key pair (ask, apk), where ask is the private key and apk the corresponding public key.

Adjudication Setup (optional): $\mathsf{AdjSetup}(ask, pk)$, on input is the private key of the adjudicator ask and a (partial) public key $\widetilde{pk}$ of the signer, returns a key pk'.

Key Generation: $\mathsf{Kg}^{\mathsf{AdjSetup}(ask,\cdot)}(1^n)$ may interact with the adjudicator via the optional oracle $\mathsf{AdjSetup}(ask,\cdot)$ to produce the key pair (sk, pk).

Signing and Verification: Same as in DS.

VES Creation: $\mathsf{Create}(sk, apk, msg)$ takes as input a secret key sk, the adjudicator's public key apk, and a message $msg \in \mathcal{M}$ from the message space $\mathcal{M}$. It returns a verifiably encrypted signature ϖ.

VES Verification: $\mathsf{VesVf}(apk, pk, \varpi, msg)$ takes as input the adjudicator's public key apk, a public key pk, a verifiably encrypted signature ϖ, and a message msg. It returns a bit.

Adjudication: $\mathsf{Adj}(ask, pk, \varpi, msg)$ takes as input the key pair (ask, apk) of the adjudicator, the public key of the signer pk, a verifiably encrypted signature ϖ, and a message msg. It extracts an ordinary signature σ for msg.

A VES scheme is *complete* if for all adjudication key pairs $(ask, apk) \leftarrow \mathsf{AdjKg}(1^n)$ and for all signature key pairs $(sk, pk) \leftarrow \mathsf{Kg}^{\mathsf{AdjSetup}(ask,\cdot)}(1^n)$ the following holds: $\mathsf{VesVf}(apk, pk, \mathsf{Create}(sk, apk, msg), msg) = 1$ and $\mathsf{Vf}(pk, \mathsf{Adj}(ask, apk, pk, \mathsf{Create}(sk, apk, msg)), msg) = 1$ for all $msg \in \mathcal{M}$.

Discussion Note that the regitration algorithm $\mathsf{AdjSetup}$ only takes as input the signer's public key and *not* the private key. Thus, this phase cannot be compared with other key-registration models that require the signer to prove knowledge of the secret key. Moreover, this phase only takes place once, during key generation and *not* during each signature creation process. The adjudicator remains offline, i.e., our modification is suitable for fair exchange protocols with a *passive* adjudicator. Via $\mathsf{AdjSetup}$, the adjudicator may define parts of signer keys. Giving the adjudicator too much control over, however, is discouraged as it affects non-frameability.

5.2 A New Security Model

The security requirements for verifiably encrypted signatures have undergone a series of changes since their introduction in [BGLS03]. In their original model, a VES scheme needs to satisfy unforgeability and opacity. *Unforgeability* requires that it is hard to forge a verifiably encrypted signature and *opacity* implies that it is difficult to extract an ordinary signature from a verifiably encrypted signature.

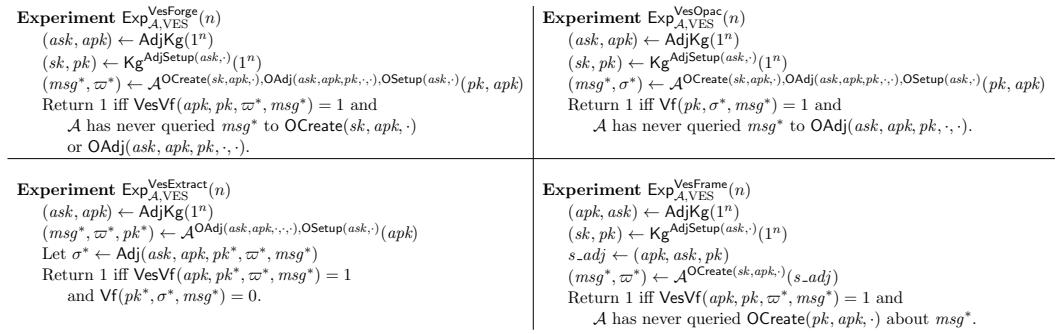


Figure 5.1: Overview over the different security experiments for VES.

In [RS09], we discover a flaw in the original security model and propose to extend it with extractability and non-frameability[1]. *Extractability* guarantees that the adjudicator can always extract a regular signature from a valid verifiably encrypted signature. This property should even hold for adversely generated signing keys. *Non-frameability* prevents signer and adjudicator from successfully colluding in order to produce a verifiably encrypted signature on behalf of another party, provided that the collusion happens in the online phase and *not* during key registration. The security requirement can be interpreted as a stronger form of unforgeability, which we will prove in Section 5.2.3. The two additional requirement are justified in Section 5.2.1 and Section 5.2.2, respectively.

Unforgeability and opacity are formalized in experiments $\mathsf{Exp}^{\mathsf{VesForge}}_{\mathcal{A},\mathrm{VES}}$ and $\mathsf{Exp}^{\mathsf{VesOpac}}_{\mathcal{A},\mathrm{VES}}$, where the adversary is given the public keys of the signer and of the adjudicator. Moreover, the adversary has access to two oracles: $\mathsf{OCreate}$ returns verifiably encrypted signatures for a given message and OAdj extracts a regular signature from a verifiably encrypted signature. In the extractability experiment $\mathsf{Exp}^{\mathsf{VesExtract}}_{\mathcal{A},\mathrm{VES}}$, the adversarial signer is given access to an adjudication oracle and wins if he or she can output an encrypted signature that is hidden irrecoverably. Here, the adversary is also allowed to pick its own signing key. Finally, the non-frameability experiment $\mathsf{Exp}^{\mathsf{VesFrame}}_{\mathcal{A},\mathrm{VES}}$ gives the adversary direct access to the adjudicator's private key. The goal is to forge a signature for another party. When using the optional registration algorithm $\mathsf{AdjSetup}$ in Kg, we also give the adversary access to the corresponding oracle in all experiments. All experiments are defined in Figure 5.1.

Definition 5.1 (Security of VES)**.** *VES is* secure *if it is unforgeable, opaque, extactable, and non-frameable according to the following definitions, where $\mathcal{A}$ is an*

[1]Previously referred to as "collusion-resistance" or "abuse-freeness" in [RS09, RSS10].

efficient adversary.

Unforgeability: *VES is* unforgeable *if* $\mathsf{Exp}^{\mathsf{VesForge}}_{\mathcal{A},VES}(n)$ *outputs 1 with negligible probability.*

Opacity: *VES is* opaque *if* $\mathsf{Exp}^{\mathsf{VesOpac}}_{\mathcal{A},VES}(n)$ *outputs 1 with negligible probability.*

Extractability: *VES is* extractable *if* $\mathsf{Exp}^{\mathsf{VesExtract}}_{\mathcal{A},VES}(n)$ *outputs 1 with negligible probability.*

Non-frameability: *VES is* non-frameable *if* $\mathsf{Exp}^{\mathsf{VesFrame}}_{\mathcal{A},VES}(n)$ *outputs 1 with negligible probability.*

Notice that AdjSetup and the setup oracle OSetup are always controlled by the experiments in Figure 5.1 to make the setup procedure trusted. A straightforward extension of our security model would be to remove this trusted setup procedure.

5.2.1 The Need for Extractability

We have proposed "extractability" as a new security requirement. Basically, it states that if ϖ is valid under VesVf, the adjudicator must always be able to extract an ordinary signature that is valid under Vf for an adversely chosen verification key.

Here, we motivate the need for this new property, showing that every verifiably encrypted signature scheme, secure in the model of [BGLS03], can simply be turned into one which is not extractable.

Proposition 5.2. *Let* $VES = (\mathsf{AdjKg}, \mathsf{AdjSetup}, \mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf}, \mathsf{Adj})$ *be unforgeable and opaque. Then, there exists an unforgeable and opaque scheme* VES' *that is not extractable.*

The basic idea is that the verifiably encrypted signature may consist of two *independent* parts. One part is the encrypted signature and the other part forms a "proof" of its validity. As both parts are independent, a malicious signer can easily set the encrypted signature to an empty string while computing the proof honestly.

Proof. Let $|\varpi|$ denote the bit length of a verifiably encrypted signature in VES. Our modified scheme VES′ is defined as follows.

Key Generation, Adjudication Setup (optional), Signing, Verification: Same as in VES.

VES Creation: Given a message msg, a signing key sk, and the public key of the adjudicator apk. Create' computes $\varpi' \leftarrow \mathsf{Create}(sk, apk, msg)$ and outputs $(\varpi_1 \| \varpi_2) \leftarrow (\varpi' \| \varpi') \in \{0,1\}^{2|\varpi|}$.

VES Verification: Given a verifiably encrypted signature $\varpi_1 \| \varpi_2$ on msg, VesVf' outputs 1 if and only if $\mathsf{VesVf}(apk, pk, \varpi_1, msg) = 1$.

Adjudication: $\mathsf{Adj}'(ask, pk, \varpi_1 \| \varpi_2, msg)$ outputs $\sigma \leftarrow \mathsf{Adj}(ask, pk, \varpi_2, msg)$.

Obviously, if VES is complete, unforgeable, and opaque, so is VES′. Although, now, the following adversary $\mathcal{A}$ contradicts extractability with probability 1.

Setup: $\mathcal{A}$ receives the adjudicator's public key apk and honestly generates its signature key pair $(sk, pk) \leftarrow \mathsf{Kg}^{\mathsf{AdjSetup}(ask,\cdot)}(1^n)$.

VES Creation: When $\mathcal{A}$ signs a message msg, it calls $\varpi_1 \| \varpi_2 \leftarrow \mathsf{Create}'(sk, apk, msg)$ and outputs $(msg^*, \varpi^*, pk^*) \leftarrow (msg, \varpi_1 \| 0^{|\varpi|}, pk)$.

Since ϖ_1 remains unchanged in Create', VesVf' always returns 1. The algorithm Adj', however, cannot extract a valid (ordinary) signature from ϖ_2 because it is the 0-string. □

5.2.2 The Need for Non-frameability

In order to justify the need for non-frameability, we prove the following separation from the model due to Boneh et al.

Proposition 5.3. *Let* $VES = (\mathsf{AdjKg}, \mathsf{AdjSetup}, \mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf}, \mathsf{Adj})$ *be unforgeable and opaque, let* D *be the space of secret signer keys in VES and let* $TFF = (\mathsf{Kg}, \mathsf{Eval}, \mathsf{Inv})$ *be a family of trapdoor functions with domain* D*. Then, there exists an unforgeable and opaque scheme* VES' *that is not non-frameable.*

The idea of the proof is as follows. We build a verifiably encrypted signature scheme VES′ out of VES such that VES′ remains unforgeable and opaque but such a malicious adjudicator is able to reveal the private signing key. Hence, if $\mathcal{A}$ colludes with the adjudicator in $\mathsf{Exp}^{\mathsf{VesFrame}}_{\mathcal{A},\mathrm{VES}}$, it can learn the secret signing key.

Proof. The algorithms in VES′ are the same as in VES with the following changes.

Adjudication Key Generation: $\mathsf{AdjKg}'(1^n)$ generates $(ask, apk) \leftarrow \mathsf{AdjKg}(1^n)$ and selects a key-pair for the trapdoor function $(s,t) \leftarrow \mathrm{TFF}.\mathsf{Kg}(1^n)$. It outputs $(ask', apk') \leftarrow ((ask, t), (apk, s))$.

VES Creation: $\mathsf{Create}'(sk, (apk, s), msg)$ executes the underlying creation algorithm $\varpi' \leftarrow \mathsf{Create}(sk, apk, msg)$ and then computes $a \leftarrow \mathrm{TFF}.\mathsf{Eval}(s, sk)$. It outputs the verifiably encrypted signature $\varpi' \leftarrow (\varpi, a)$.

VES Verification: $\mathsf{VesVf}'((apk, s), pk, (\varpi, a), msg)$ outputs the result of the underlying verification algorithm $\mathsf{VesVf}(apk, pk, \varpi, msg)$.

Adjudication: $\mathsf{Adj}'((ask, t), pk, (\varpi, a), msg)$ outputs the result of the underlying adjudication algorithm $\mathsf{Adj}(ask, pk, \varpi, msg)$.

Obviously, the resulting scheme is still unforgeable and opaque if TFF is hard to invert without t (cf. $\mathsf{Exp}^{\mathsf{TRAP\text{-}OW}}_{\mathcal{A},\mathrm{TFF}}$ in Section 2.3.2). However, the adversary in $\mathsf{Exp}^{\mathsf{VesFrame}}_{\mathcal{A},\mathrm{VES}}$ can easily frame any signer with probability 1 because it has access to the trapdoor t. More precisely, it can query the target signer via $\mathsf{OCreate}$, obtain a verifiably encrypted signature (ϖ, a), and recover sk from a. Then, given sk, it can sign any message on behalf of the target signer. □

5.2.3 Implications

The purpose of this section is to provide two propositions that simplify the security proofs for VES schemes. The first shows that non-frameability implies unforgeability.

Proposition 5.4. *If VES is non-frameable, it is also unforgeable.*

To see this, observe that giving ask to $\mathcal{A}$ in $\mathsf{Exp}^{\mathsf{VesFrame}}_{\mathcal{A},\mathrm{VES}}$ enables the adversary to simulate the oracles OSetup and OAdj itself. Thus, if $\mathcal{A}$ is successful in $\mathsf{Exp}^{\mathsf{VesForge}}_{\mathcal{A},\mathrm{VES}}$, with *only* oracle access to OSetup and OAdj, it can also break non-frameability.

The second simplification, showing that extractability implies non-frameability, applies whenever the VES scheme in question follows a common construction principle. An obvious design choice for VES schemes is to employ a signature scheme DS in a black-box fashion to sign messages, then apply an encryption scheme to hide the ordinary signature, and append some form of proof for verifiability. All previous constructions, e.g., [BGLS03, LOS+06], adhere to this concept, which we formalize in the following definition.

Definition 5.5 (Key-independence)**.** *Let* $DS = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ *be a digital signature scheme. In VES, let the signer's private key sk consist of two independent elements* $(kisk, ssk)$ *and let* $pk = (kipk, spk)$ *be the corresponding public key. VES is* key-independent *if there is an efficient "encryption" algorithm* $\mathsf{KI\text{-}Enc}$ *such that* Create *works in three steps:*

1. $\sigma \leftarrow DS.\mathsf{Sign}(ssk, msg)$;
2. $\varpi \leftarrow \mathsf{KI\text{-}Enc}(apk, kipk, kisk, \sigma, msg)$;
3. *Output* ϖ.

Observe that the algorithm KI-Enc does not have access to the signing key ssk in DS. Hence, in reduction proofs, we can easily replace step 1 with an external signature oracle. In this setting, we show that extractability implies non-frameability and, therefore, also unforgeability. In consequence, proving security of key-independent VES schemes only involves proving extractability and opacity.

Proposition 5.6. *Let VES be extractable and key-independent with an EU-CMA secure signature scheme DS. Then, VES is non-frameable.*

Here, the intuition is to replace the signature algorithm with a signature oracle from the $\mathsf{Exp}^{\mathsf{EU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}$ experiment for DS and to honestly simulate the adjudicator. Since extractability guarantees that every valid verifiably encrypted signature yields a valid ordinary signature, a forgery in the sense of $\mathsf{Exp}^{\mathsf{VesFrame}}_{\mathcal{A},\mathrm{VES}}$ can be converted into a forgery against DS.

Let us now briefly prove the above propositions.

Proof of Proposition 5.4. Towards contradiction, let $\mathcal{A}$ be a successful, efficient adversary against unforgeability of VES = (AdjKg, AdjSetup, Kg, Sign, Vf, Create, VesVf, Adj). Given black-box access to $\mathcal{A}$, we construct an efficient algorithm $\mathcal{B}$ that refutes the assumed non-frameability.

$\mathcal{B}$ takes as input $s\_adj = (apk, ask, pk)$ and runs $\mathcal{A}(pk, apk)$. Whenever $\mathcal{A}$ queries OCreate, the query if forwarded to $\mathcal{B}$'s own oracle. Whenever $\mathcal{A}$ queries OSetup or OAdj, $\mathcal{B}$ honestly simulates these oracles with ask. Eventually, $\mathcal{A}$ stops and outputs a forgery (msg^*, ϖ^*) such that $\mathsf{VesVf}(apk, pk, \varpi^*, msg^*)$ and such that it has never queried msg^* to OCreate. Hence, $\mathcal{B}$ can forward this forgery and wins in $\mathsf{Exp}^{\mathsf{VesFrame}}_{\mathcal{A},\mathrm{VES}}$. □

Proof of Proposition 5.6. Let VES = (AdjKg, AdjSetup, Kg, Sign, Vf, Create, VesVf, Adj) be extractable and let DS = (Kg, Sign, Vf) be the underlying EU-CMA secure signature scheme. Suppose that there is an efficient adversary $\mathcal{A}$ that successfully breaks non-frameability.

We construct an algorithm $\mathcal{B}^{\mathcal{A}}$, that breaks unforgeability of DS. $\mathcal{B}$ receives a public verification key spk and has access to a signing oracle $\mathsf{OSign}(ssk, \cdot)$. It generates an adjudication key pair $(ask, apk) \leftarrow \mathsf{AdjKg}(1^n)$ and runs the remaining part of VES.Kg, including AdjSetup, to obtain a VES key pair (sk, pk) =

$((kisk, \emptyset), (kipk, spk))$. This is possible because VES is key-independent. Afterwards, $\mathcal{B}$ sets $s\_adj \leftarrow (ask, apk, pk)$ and runs $\mathcal{A}(s\_adj)$ as a black-box. Whenever $\mathcal{A}$ queries a message msg to $\mathsf{OCreate}$, $\mathcal{B}$ calls its external signing oracle $\sigma \leftarrow \mathsf{OSign}(ssk, msg)$ and computes $\varpi \leftarrow \mathsf{KI\text{-}Enc}(apk, kipk, kisk, \sigma, msg)$. Eventually, $\mathcal{A}$ stops and outputs (msg^*, ϖ^*). $\mathcal{B}$ extracts the corresponding signature $\sigma^* \leftarrow \mathsf{Adj}(ask, pk, \varpi^*, msg^*)$ and returns (msg^*, σ^*).

Observe that the environment of $\mathcal{A}$ is perfectly simulated and all oracle queries are answered efficiently because VES is key-independent. By definition, $\mathcal{A}$ has not queried msg^* to $\mathsf{OCreate}$. Thus, $\mathcal{B}$ has not queried msg^* to its signature oracle. Moreover, the resulting (msg^*, ϖ^*) yields an ordinary message-signature pair (msg^*, σ^*) because VES is extractable. As a consequence, $\mathcal{B}$'s attack is legitimate and it succeeds in $\mathsf{Exp}^{\mathsf{EU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}$. □

5.3 Generic Constructions

We propose two generic constructions based on passively-secure encryption, digital signature schemes, and collision-resistant hash functions. Both constructions are key-independent and they use what we define as "maskable" signatures in conjunction with a weaker form of CPA encryption to hide an ordinary signature. The employed encryption scheme is completely independent of the underlying signature scheme. In particular, they need not operate on the same objects. Verifiability is ensured by the maskability of the signature scheme and in order to ensure extractability, we build a Merkle authentication tree from a collision-resistant hash function. It will provide a link between the maskable signature scheme and the encryption scheme. While our first construction uses regular signature schemes, our second construction reduces the assumptions even further by merely relying on one-time signatures.

5.3.1 Building Blocks

In the following, we define and discuss the ingredients for our constructions.

Random Plaintext Secure Encryption Let PKE = $(\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ be a public key encryption scheme. We define a new notion of security for encryption schemes that we call security under *random plaintext attacks* (RPA). It is weaker than CPA. The idea is that the adversary obtains a randomly chosen message msg and a ciphertext ct. The task is to determine whether ct encrypts msg or the constant 0-string.

Definition 5.7 (RPA Encryption)**.** *PKE is indistinguishable under random plaintext attacks (RPA) if no efficient algorithm $\mathcal{A}$ can associate a randomly generated plaintext with its ciphertext. This is formalized in the following experiment* $\mathsf{Exp}^{RPA}_{\mathcal{A},PKE}$*.*

Experiment $\mathsf{Exp}^{RPA}_{\mathcal{A},PKE}(n)$
$(esk, epk) \leftarrow \mathsf{Kg}(1^n)$
$b \leftarrow_{\$} \{0,1\}$
$msg_0 \leftarrow_{\$} \mathcal{M}$
$msg_1 \leftarrow_{\$} 0^{|msg_0|}$
$ct^* \leftarrow \mathsf{Enc}(epk, msg_b)$
$d \leftarrow \mathcal{A}(epk, ct^*)$
Return 1 *if and only if* $d = b$*.*

We claim that RPA is strictly weaker than CPA, in the sense that any CPA scheme is also RPA, but not vice-versa.

Proposition 5.8. *A CPA secure scheme is also RPA secure. If an RPA secure scheme, encrypting $\omega(\log(n))$ bits, exists then there is also an RPA secure scheme that is not CPA secure.*

Proof (Sketch). The first part is obvious because an adversary against CPA has full control over the messages. As for the second part, let $\mathrm{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ be an RPA secure encryption scheme with message length $\kappa = \omega(\log(n))$. The basic idea is to modify the scheme and let Enc output a fixed, publicly known ciphertext ct^* when queried with $1\|0^{\kappa-1}$ and actually encrypt otherwise. The scheme is still RPA secure because $msg_0 \neq 1\|0^{\kappa-1}$ but with negligible probability. However, it is clearly not CPA secure because an adversary in $\mathsf{Exp}^{\mathsf{CPA}}_{\mathcal{A},\mathrm{PKE}}$ can specifically submit $1\|0^{\kappa-1}$ as one of its messages and look for ct^* in the output. □

Maskable Signature Schemes For our generic construction, we need a signature scheme that is "maskable". Generally speaking, it means that a signature can be hidden using a masking value, such that we can still verify it. At the same time, it must be hard to recover the signature without knowing the masking value. We formalize this in the following definition.

Definition 5.9 (Maskability)**.** *Let $DS = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ be a signature scheme with public-key space $\mathcal{K}$, signature space Σ, and message space $\mathcal{M}$. It is* maskable *if there*

is a corresponding masking scheme $MS_{DS} = (\mathsf{Advice}, \mathsf{Mask}, \mathsf{Unmask}, \mathsf{Vf}, \Psi)$ *with the following specification.*

Sets: *Let* $\mathcal{S}$ *be a set of masking values and let* Ψ *be a distribution over* $\mathcal{S}$*. We write* $x \leftarrow_\Psi S$ *when* x *is chosen from* Ψ *over* S*. Furthermore, let* $\mathcal{V}$ *be the set of advice strings for verifying masked signatures from the space* $\mathcal{T}$ *of such signatures.*

Advice: $\mathsf{Advice}(spk, \alpha)$*, on input* $spk \in \mathcal{K}$ *and* $\alpha \in \mathcal{S}$*, outputs an advice string* $\beta \in \mathcal{V}$*.*

Mask: $\mathsf{Mask}(spk, \sigma, \alpha, msg)$*, on input* $spk \in \mathcal{K}$*,* $\sigma \in \Sigma$*,* $\alpha \in \mathcal{S}$*, and* $msg \in \mathcal{M}$*, outputs a masked signature* $\tau \in \mathcal{T}$*. Notice that we do not require a perfect masking scheme but allow the scheme to output the special symbol* $\perp$ *if masking fails.*

Unmask: $\mathsf{Unmask}(\tau, \beta, \alpha, msg)$*, on input* $\tau \in \mathcal{T}$*,* $\beta \in \mathcal{V}$*,* $\alpha \in \mathcal{S}$*, and* $msg \in \mathcal{M}$*, outputs a signature* $\sigma \in \Sigma$*.*

Verification: $MS_{DS}.\mathsf{Vf}(spk, \tau, \beta, msg)$*, on input* $spk \in \mathcal{K}$*,* $\tau \in \mathcal{T}$*,* $\beta \in \mathcal{V}$*, and* $msg \in \mathcal{M}$*, outputs a bit, indicating the validity of the masked signature. If* $\tau = \perp$*, it returns* 0*.*

It is complete *if an honestly masked signature can be successfully verified under* $MS_{DS}.\mathsf{Vf}$ *with a probability bounded away from zero. Regarding its security, we require that* MS_{DS} *is binding and hiding.*

Binding: *We require that it is hard for an adversary to produce a masked signature that cannot be unmasked to obtain a regular (DS) signature, even if the verification key of DS is chosen by the adversary. We call this requirement* binding *and it is formalized in the following experiment* $\mathsf{Exp}^{\mathit{MaskBind}}_{\mathcal{A},MS}$*, where the adversary* $\mathcal{A}$ *works in two modes ("find" and "forge"). In mode "find",* $\mathcal{A}$ *outputs a signature verification key for DS. Then, the experiment chooses a masking value and runs* $\mathcal{A}$ *in mode "forge". Furthermore,* $\mathcal{A}$ *is allowed to keep a state across modes. The adversary wins if it outputs a masked signature* τ^* *for a message* msg^* *that is correct according to* $MS_{DS}.\mathsf{Vf}$ *but for which* Unmask *fails to produce a valid DS signature.*

Experiment $\mathsf{Exp}^{\mathit{MaskBind}}_{\mathcal{A},MS}(n)$
$(spk^*, s\_find) \leftarrow \mathcal{A}(m\_find)$
$\alpha \leftarrow_\Psi \mathcal{S}$
$\beta \leftarrow \mathsf{Advice}(spk^*, \alpha)$
$(msg^*, \tau^*) \leftarrow \mathcal{A}(m\_forge, s\_find, \alpha)$
$\sigma^* \leftarrow \mathsf{Unmask}(\tau^*, \beta, \alpha, msg^*)$
Return 1 $DS.\mathsf{Vf}(spk, \sigma^*, msg^*) = 0$ *and* $MS_{DS}.\mathsf{Vf}(spk^*, \tau^*, \beta, msg^*) = 1$

Hiding: *In addition, a masked signature should be hard to unmask without knowledge of the masking value. This must even hold if the adversary can query an oracle* OMask once *that returns a masked signature for an adversely chosen message and a randomly chosen* α*:* $\mathsf{OMask}(ssk, spk, \alpha, msg) = [\sigma \leftarrow DS.\mathsf{Sign}(ssk, msg); \tau \leftarrow \mathsf{Mask}(spk, \sigma, \alpha, msg); \text{Return } \tau;]$*. Furthermore, the adversary can make arbitrary queries to an ordinary signature oracle. This is formalized in the experiment* $\mathsf{Exp}^{\mathit{UnMask}}_{\mathcal{A},MS}$*. The adversary wins, if it is able to output a signature for the message that has been queried to* OMask *and not to the signature oracle.*

Experiment $\mathsf{Exp}^{\mathit{UnMask}}_{\mathcal{A},MS}(n)$
$(ssk, spk) \leftarrow DS.\mathsf{Kg}(1^n)$
$\alpha \leftarrow_\Psi \mathcal{S}$
$\beta \leftarrow \mathsf{Advice}(spk, \alpha)$
$\sigma^* \leftarrow \mathcal{A}^{\mathsf{OMask}(ssk, spk, \alpha, \cdot), \mathsf{OSign}(ssk, \cdot)}(spk, \beta)$
Let $(msg_i)_{i=1}^{\ell}$ *be the queries to* $DS.\mathsf{Sign}$*.*
Let msg^* *be the query to* OMask*.*
Return 1 *iff* $msg^* \notin (msg_i)_{i=1}^{\ell}$ *and* $DS.\mathsf{Vf}(spk, \sigma^*, msg^*) = 1$*.*

Notice that the above definition can be satisfied via straightforward encryption in combination with letting Advice output a NIZK proof proof. We propose that somewhat homomorphic signature schemes can provide the same functionality as illustrated in the following example.

Example 5.10. *Consider the RSA signature scheme with full-domain hash function* H *and public key* (N, v) *[BR96]. The verification algorithm for a signature* σ *on a message msg checks whether* $0 < \sigma < N$ *and* $\sigma^v \equiv \mathsf{H}(msg) \pmod N$*. We let* $\Sigma = \mathcal{V} = \mathbb{Z}_N$ *and* $\mathcal{S} = \mathbb{Z}_N^*$*.* Ψ *is the uniform distribution.* $\mathsf{Mask}((N, v), \sigma, \alpha, msg)$ *outputs*

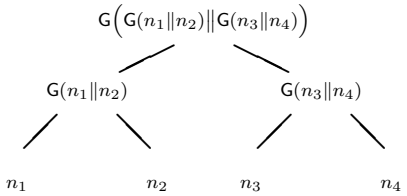


Figure 5.2: Merkle tree with leaf values n_1, n_2, n_3, n_4.

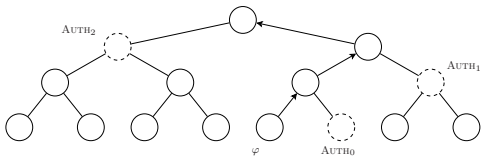


Figure 5.3: The authentication path of leaf φ.

$\sigma\alpha \bmod N$ *and* $\mathsf{Advice}((N, v), \alpha)$ *returns* $\alpha^v \bmod N$. *Thus,* $\mathsf{Unmask}(\tau, \beta, \alpha, msg)$ *has to compute* $\sigma \leftarrow \tau\alpha^{-1} \bmod N$. *The modified verification algorithm MS.*$\mathsf{Vf}(spk, \tau, \beta, msg)$ *checks whether* $0 < \tau < N$ *and* $\tau^v \equiv \mathsf{H}(msg)\beta$. *Observe that the scheme is binding and hiding in the random oracle model. For the full description of an entirely RSA-based construction, refer to [Rüc09].*

Given the above example, it is easy to see that one can forge a masked signature (τ, β) that passes MS.Vf, unless α and $\beta = \mathsf{Advice}(pk, \alpha)$ are well-formed. One could simply compute $\beta \leftarrow \tau^v / \mathsf{H}(msg)$ for arbitrary msg and τ. The result (τ, β, msg) would be valid because $\tau^v \equiv \mathsf{H}(msg)\beta$. However, in our constructions, the attacker will not be able to choose β freely. It is chosen during key registration and then authenticated with a hash tree. This authentication mechanism yields an implicit rejection of adversely chosen β.

Merkle Authentication Trees In [Mer89], Merkle proposes a tree-based authentication mechanism for large amounts of data using only a single hash value. His idea has led to, e.g., the construction of digital signature schemes out of one-time signature schemes and hash functions. With our constructions, we add verifiable encryption to the list of potential applications.

A Merkle tree is a complete binary tree of height h that is built from the bottom up to the root such that the set of leaves defines the entire tree. The leaves are numbered consecutively from left to right. Inner nodes are constructed using the following rule: a node's value is the hash value of the concatenation of its children *left* and *right*: node $= \mathsf{G}(left\|right)$, where $\mathsf{G} : \{0,1\}^* \to \{0,1\}^n$ is a collision-resistant hash function. See Figure 5.2 for an example. The root ρ of the tree is used to authenticate the leaf values. For the authentication process, additional tree nodes are required. These nodes form the *authentication path* of a leaf. Consider the path from the leaf with index $\varphi \in [2^h]$ to the root. The siblings of the nodes on this path form the authentication path π_φ of this leaf (cf. Figure 5.3 for an example).

Given a leaf node η_φ, its authentication path π_φ, and the root ρ, one can rebuild the entire tree with the construction rule $\mathsf{G}(\mathit{left}\|\mathit{right})$. If the calculated root matches ρ, the leaf η_φ is correctly authenticated.

An adversary that is able to replace a leaf value, such that the replaced leaf is still correctly authenticated in the tree, is also able to find collisions in the underlying hash function G. For an overview of techniques, potential efficiency improvements, and references, we refer the reader to [BDS08].

5.3.2 Construction 1

The general idea is to use a maskable signature scheme with one-time masking values and encrypt these masking values under the adjudicator's public key. The *binding* property of the masking scheme ensures completeness and opacity will be guaranteed by the *hiding* property. We take an ordinary signature σ and hide it by applying Mask, using one of ℓ predefined one-time masking values α. If, for any reason, the masking scheme returns an invalid masked signature, the process is repeated with the next α. This allows for a broader range of (imperfect) masking schemes. The corresponding advice β for verification is also precomputed. Then, β and an encryption γ of α are used to build a Merkle authentication tree that allows a verifier to efficiently check whether β and γ correspond. The adjudicator forms the tree during the initial registration phase in $\mathsf{AdjSetup}$ and signs its root under a certification key pair $(\mathit{csk}, \mathit{cpk})$ in order to prevent malicious signers from cheating in the extractability experiment.

Let DS be a maskable signature scheme with masking scheme $\mathrm{MS}_{\mathrm{DS}}$, PKE be a public key encryption scheme and $\mathsf{G} : \{0,1\}^* \mapsto \{0,1\}^n$ be a collision-resistant hash function. Choose an adequate $h \in \mathbb{N}$, such that the resulting scheme admits $\ell = 2^h = \mathrm{poly}(n)$ signatures. $\mathrm{VES}_1 = (\mathsf{AdjKg}, \mathsf{AdjSetup}, \mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf}, \mathsf{Adj})$ is defined as follows.

Adjudicator Key Generation: $\mathsf{AdjKg}(1^n)$ calls $(\mathit{esk}, \mathit{epk}) \leftarrow \mathrm{PKE}.\mathsf{Kg}(1^n)$, $(\mathit{csk}, \mathit{cpk}) \leftarrow \mathrm{DS}.\mathsf{Kg}(1^n)$, and outputs $(\mathit{ask}, \mathit{apk}) \leftarrow ((\mathit{esk}, \mathit{csk}), (\mathit{epk}, \mathit{cpk}))$.

Adjudication Setup: $\mathsf{AdjSetup}((\mathit{esk}, \mathit{csk}), \mathit{spk})$ performs the following steps:

1. Choose $\alpha_i \leftarrow_\Psi \mathcal{S}$ and set $\beta_i \leftarrow \mathsf{Advice}(\mathit{spk}, \alpha_i)$, $\gamma_i \leftarrow \mathsf{Enc}(\mathit{epk}, \alpha_i)$ for $i = 1, \ldots, \ell$;
2. Construct a Merkle tree T using G, i.e., with leaves $\mathsf{G}\Big(\mathsf{G}(\beta_i)\|\mathsf{G}(\gamma_i)\Big)$ that fully define the root ρ;
3. Compute the signature $\sigma_\rho \leftarrow \mathrm{DS}.\mathsf{Sign}(\mathit{csk}, \rho)$;

4. Output $((\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell}, \rho, \sigma_\rho)$.

Key Generation: $\mathsf{Kg}^{\mathsf{AdjSetup}(ask,\cdot)}(1^n)$ performs the following steps:

1. Call $(ssk, spk) \leftarrow \mathrm{DS}.\mathsf{Kg}(1^n)$;
2. Call $((\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell}, \rho, \sigma_\rho) \leftarrow \mathsf{AdjSetup}(ask, spk)$;
3. Initialize a signature counter $c \leftarrow 0$;
4. Output $pk = (spk, \rho, \sigma_\rho)$ and $sk = (ssk, c, (\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell})$.

Sign, Verify: As defined in the underlying signature scheme DS.

VES Creation: $\mathsf{Create}((ssk, c, (\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell}), apk, msg)$ works in three steps:

1. Increment the counter c: $c \leftarrow c + 1$;
2. Sign msg using the underlying signature scheme: $\sigma \leftarrow \mathrm{DS}.\mathsf{Sign}(ssk, msg)$;
3. Mask σ with the secret value α_c: $\tau \leftarrow \mathsf{Mask}(spk, \sigma, \alpha_c, msg)$;
4. If $\mathrm{MS}_{\mathrm{DS}}.\mathsf{Vf}(spk, \tau, \beta_c, msg) = 0$ increase c and go to 3.

The output is $\varpi = (\tau, \beta_c, \gamma_c, \pi_c)$, where $\beta_c \leftarrow \mathsf{Advice}(spk, \alpha_c)$ and π_c is the authentication path for leaf c.

VES Verification: $\mathsf{VesVf}((epk, cpk), (spk, \rho, \sigma_\rho), (\tau, \beta, \gamma, \pi), msg)$ outputs 1 iff

1. $\mathrm{DS}.\mathsf{Vf}(cpk, \sigma_\rho, \rho) = 1$;
2. π is correct for β and γ with respect to ρ;
3. $\mathrm{MS}_{\mathrm{DS}}.\mathsf{Vf}(spk, \tau, \beta, msg) = 1$.

Adjudication: $\mathsf{Adj}((esk, csk), pk, (\tau, \beta, \gamma, \pi), msg)$ verifies the input using VesVf. If it is correct, it decrypts $\alpha' \leftarrow \mathsf{Dec}(esk, \gamma)$, calls $\sigma' \leftarrow \mathsf{Unmask}(\tau, \beta, \alpha', msg)$, and outputs σ'.

5.3.3 Construction 2

Since we already need a Merkle authentication tree for our first construction, we can as well use a suitable one-time signature instead of a regular one. One-time signature schemes are potentially easier to achieve, i.e., they may be secure under milder assumptions. The following construction demonstrates that the second tree, which would be required to turn a one-time signature scheme into a "many-time" signature scheme, can be easily merged with the first one.

With OTS we denote a maskable one-time signature scheme with masking scheme $\mathrm{MS}_{\mathrm{OTS}}$. We define $\mathrm{VES}_2 = (\mathsf{AdjKg}, \mathsf{AdjSetup}, \mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf}, \mathsf{Adj})$ as follows.

Adjudicator Key Generation: $\mathsf{AdjKg}(1^n)$ calls $(esk, epk) \leftarrow \mathrm{PKE}.\mathsf{Kg}(1^n)$, $(csk, cpk) \leftarrow \mathrm{DS}.\mathsf{Kg}(1^n)$, and outputs $(ask, apk) \leftarrow ((esk, csk), (epk, cpk))$.

Adjudication Setup: $\mathsf{AdjSetup}(ask, (spk_i)_{i=1}^{\ell})$ performs the following steps:

1. Choose $\alpha_i \leftarrow_{\Psi} \mathcal{S}$ and set $\beta_i \leftarrow \mathsf{Advice}(spk, \alpha_i)$, $\gamma_i \leftarrow \mathsf{Enc}(epk, \alpha_i)$ for $i \in [\ell]$;
2. Construct a Merkle authentication tree T using the hash function G, where the leaves are of the form $\mathsf{G}\Big(\mathsf{G}(\beta_i) \| \mathsf{G}(\gamma_i) \| \mathsf{G}(spk_i)\Big)$. Denote the root node with ρ;
3. Compute the signature $\sigma_\rho \leftarrow \mathrm{DS}.\mathsf{Sign}(csk, \rho)$;
4. Output $((\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell}, \rho, \sigma_\rho)$.

Key Generation: $\mathsf{Kg}^{\mathsf{AdjSetup}(ask,\cdot)}(1^n)$ performs the following steps:

1. Call $(ssk_i, spk_i) \leftarrow \mathrm{OTS}.\mathsf{Kg}(1^n)$ for $i = 1, \ldots \ell$;
2. Call $((\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell}, \rho, \sigma_\rho) \leftarrow \mathsf{AdjSetup}(ask, (spk_i)_{i=1}^{\ell})$;
3. Initialize a signature counter $c \leftarrow 0$;
4. Output $pk = (\rho, \sigma_\rho)$ and $sk = ((ssk_i)_{i=1}^{\ell}, (spk_i)_{i=1}^{\ell}, c, (\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell})$.

Sign, Verify: As defined in OTS.

VES Creation: $\mathsf{Create}((ssk_i)_{i=1}^{\ell}, (spk_i)_{i=1}^{\ell}, c, (\alpha_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell}, apk, msg)$ works in four steps:

1. Increment c: $c \leftarrow c + 1$;
2. Sign msg: $\sigma \leftarrow \mathrm{OTS}.\mathsf{Sign}(ssk_c, msg)$;
3. Mask σ: $\tau \leftarrow \mathsf{Mask}(spk_c, \sigma, \alpha_c, msg)$;
4. If $\mathrm{MS}_{\mathrm{DS}}.\mathsf{Vf}(spk_c, \tau, \beta_c, msg) = 0$ go to 1.

The output is $\varpi = (\tau, \beta_c, \gamma_c, \pi_c, spk_c)$, where $\beta_c \leftarrow \mathsf{Advice}(spk, \alpha_c)$ and π_c is the authentication path for leaf c.

VES Verification: $\mathsf{VesVf}((epk, cpk), (\rho, \sigma_\rho), (\tau, \beta, \gamma, \pi, spk), msg)$ outputs 1 iff

1. $\mathrm{DS}.\mathsf{Vf}(cpk, \sigma_\rho, \rho) = 1$;

2. ρ can be reconstructed using π, β, γ, and spk;
3. $\mathrm{MS}_{\mathrm{DS}}.\mathsf{Vf}(spk, \tau, \beta, msg) = 1$.

Adjudication: $\mathsf{Adj}(ask, pk, (\tau, \beta, \gamma, \pi, spk), msg)$, Adj verifies the input using VesVf. If it is correct, it decrypts $\alpha' \leftarrow \mathsf{Dec}(ask, \gamma)$, calls $\sigma' \leftarrow \mathsf{Unmask}(\tau, \beta, \alpha', msg)$, and outputs σ'.

5.3.4 Security Proofs

We show that VES_1 satisfies the desired security requirements. Security of VES_2 is proven analogously. We prove extractability and opacity; unforgeability and non-frameability follow from Section 5.2.3.

Theorem 5.11 (Extractability)**.** *VES_1 (VES_2) is extractable if DS is unforgeable, MS_{DS} is binding, and G is a collision-resistant hash function.*

Proof. The main reduction plays against unforgeability of DS and uses the binding property of $\mathrm{MS}_{\mathrm{DS}}$ and the collision resistance of G and in the analysis. The unforgeability ensures that the adversary has to call $\mathsf{AdjSetup}$ to create the public key and binding guarantees that an extracted signature is valid if computed from an honestly masked signature. Most importantly, the collision resistance of G prevents the adversary from altering the leaves of the authentication tree, i.e., from being able to dishonestly mask a signature.

The reduction chooses the adjudication key honestly during the simulation and has access to a signature oracle for DS and to the signature verification keys spk and cpk. Thus, the adversary's environment can be perfectly, and efficiently, simulated. The adversary $\mathcal{A}$ outputs a public key $(pk^*, \rho^*, \sigma_\rho^*)$ and a pair $(msg^*, (\tau^*, \alpha^*, \gamma^*, \pi^*))$ for which VesVf outputs 1. Furthermore, we let σ' be the result of the adjudication algorithm for $(\tau^*, \beta^*, \gamma^*, \pi^*)$.

Towards contradiction, let us assume that extraction fails, i.e., $\mathrm{DS}.\mathsf{Vf}(spk, \sigma', msg^*) = 0$. From VesVf, we know that ρ^* was previously created by the simulator together with a signature σ_ρ^*, using the external signature oracle. Otherwise, we would have an existential forgery that refutes unforgeability of DS. Assume that ρ^* was formed using $(\alpha_i)_{i=1}^{\ell}, (\beta_i)_{i=1}^{\ell}, (\gamma_i)_{i=1}^{\ell}$.

VesVf guarantees that π^* is an authentication path for the leaf $\mathsf{G}(\mathsf{G}(\beta^*) \| \mathsf{G}(\gamma^*))$ w.r.t. ρ. Thus, there is an index $i \in \{1, \ldots, \ell\}$ such that $\beta^* = \beta_i = \mathsf{Advice}(spk, \alpha_i)$ and $\gamma^* = \gamma_i$. Otherwise, we would have at least one collision in the hash tree, which refutes collision resistance of G.

Finally, VesVf ensures that $\mathrm{MS}_{\mathrm{DS}}.\mathsf{Vf}(spk, \tau^*, \beta^*, msg^*) = 1$, which implies the contradiction $\mathrm{DS}.\mathsf{Vf}(spk, \sigma', msg^*) = 1$ because $\mathrm{MS}_{\mathrm{DS}}$ is binding. □

Theorem 5.12 (Opacity)**.** *VES_1 (VES_2) is opaque if DS is unforgeable, PKE is RPA secure, MS_{DS} is hiding, and G is collision-resistant.*

Proof. An adversary breaking opacity can succeed in two different ways. First, by forging the underlying signature scheme, and second, by decrypting a given verifiably encrypted signature. We say that an algorithm $\mathcal{A}$ is a

1. type-1 adversary ($\mathcal{A}_1$), if it outputs a message-signature pair (msg^*, σ^*) such that it *has never queried* msg^* to $\mathsf{OCreate}$, or if it *invokes* OAdj on msg' without having queried msg' to $\mathsf{OCreate}$ before.

2. type-2 adversary ($\mathcal{A}_2$), if it outputs a message-signature pair (msg^*, σ^*) such that it *has queried* msg^* to $\mathsf{OCreate}$ and it *has never invoked* OAdj on msg' without having queried msg' to $\mathsf{OCreate}$ before.

$\mathcal{A}_1$ can be directly used to forge signatures in DS. The reduction has control over the adjudicator's private key and can therefore extract ordinary signatures (forgeries) from $\mathcal{A}_1$'s output or query to OAdj. We omit the proof because it is straightforward.

Type-2 Attacker Here, the goal is to use $\mathcal{A}_2$ to break the hiding property of $\mathrm{MS}_{\mathrm{DS}}$. We change the simulation of $\mathcal{A}_2$'s environment and argue that each does not change $\mathcal{A}_2$'s success probability but for a negligible amount. Let $Q_{\mathsf{OCreate}} \leq \ell$ be the number of $\mathcal{A}_2$'s queries to $\mathsf{OCreate}$. First, we change the algorithm $\mathsf{AdjSetup}$.

Adjudication Setup: The algorithm $\mathsf{AdjSetup}'$ selects the elements α_i, β_i as before and chooses a random index $c^* \leftarrow_{\$} [Q_{\mathsf{OCreate}}]$. It computes all $\gamma_{i \neq c^*}$ as before but $\gamma_{c^*} \leftarrow \mathsf{Enc}(apk, 0^{|\alpha|})$, where $|\alpha|$ is the bit length of an element in $\mathcal{S}$. It outputs the corresponding leaves, root ρ, and signature σ_ρ as before.

Due to the RPA security of the encryption scheme, this only changes $\mathcal{A}_2$'s success probability by a negligible amount. The next change to $\mathsf{AdjSetup}$ allows the reduction to use $\mathcal{A}_2$ to refute the hiding property of $\mathrm{MS}_{\mathrm{DS}}$.

Adjudication Setup: The algorithm $\mathsf{AdjSetup}''$ works like $\mathsf{AdjSetup}'$, but receives β_{c^*} from the UnMask experiment and embeds it into the leaf at index c^*.

The success probability of $\mathcal{A}_2$ does not change because β_{c^*} is distributed as before. Also, knowledge of α_{c^*} is not necessary to build the modified public key.

The remaining oracles, $\mathsf{OCreate}$ and OAdj, are perfectly simulated for all indices $\neq c^*$ because the reduction has access to all masking values (except α_{c^*}) and can therefore answer all adjudication queries. In particular, this is the reason why we do not require some form of chosen-ciphertext secure encryption: all plaintexts are known and authenticated. Also, using these masking values together with the OSign and OMask oracles in the UnMask experiment, enables the reduction to answer all queries to $\mathsf{OCreate}$.

Eventually, $\mathcal{A}_2$ outputs a message-signature pair (msg^*, σ^*). If it is valid for the index c^*, the reduction outputs σ^* to refute the hiding property. Otherwise, it aborts. The reduction's success probability is noticeable if $\mathcal{A}_2$'s success probability is noticeable and the computational overhead is negligible. □

Since VES_1 and VES_2 are extractable and key-independent, unforgeability follows from Proposition 5.4 and Proposition 5.6 guarantees non-frameability.

5.4 An Instantiation with Lattices

Basically, we have already seen in Example 5.10 that our first generic construction can be instantiated from the RSA assumption in the random oracle model. A second instantiation in the random oracle model is in [RSS10]. It is based on the PSTF-based GPV signature scheme [GPV08].

In the following, we demonstrate the feasibility of our second construction. In particular, we show that it can be realized entirely based on the worst-case hardness of lattice problems in the standard model.

Building Blocks For our second construction, we require a collision-resistant hash function G, a public-key encryption scheme PKE, a one-time signature scheme OTS, and a corresponding masking scheme $\mathrm{MS}_{\mathrm{OTS}}$.

The first two ingredients already exist in lattice cryptography: we instantiate G with SWIFFTX [ADL+08], PKE with any CPA secure scheme based on LWE [Reg09, GPV08, LPR10]. This is sufficient as CPA implies RPA. Using ring-LWE [LPR10] bears the advantage of using efficiency-improving ideal lattices in all building blocks. As a result, all algorithms (except $\mathsf{Kg}^{\mathsf{AdjSetup}}$) can be implemented to run in quasi-linear time $\widetilde{\mathcal{O}}(n)$.

What is left to show is that there is a maskable one-time signature scheme from (ideal) lattices. To this end, we recall the LM-OTS one-time signature scheme [LM08].

Let LM-OTS = $(\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ with parameters $q = \text{poly}(n)$, $m = \Omega(\log(n))$, and $d = \widetilde{\mathcal{O}}(n)$. It is based on ideal lattices and uses a compression function $h = h_{\hat{\mathbf{a}}}$ for $\hat{\mathbf{a}} \leftarrow_{\$} \mathbf{R}^m$, which maps $\mathbf{R}_0^m \to \mathbf{R}$, i.e., $\hat{\mathbf{x}} \mapsto \sum_{i=1}^m \mathbf{a}_i \mathbf{x}_i \bmod q$ (cf. Section 2.7.1). Its specification is as follows.

Key Generation: $\mathsf{Kg}(1^n)$ outputs a secret key $(\hat{\mathbf{k}}, \hat{\mathbf{l}}) \in \mathbf{R}_0^m \times \mathbf{R}_0^m$ (with small norm) and a public key $(h, \mathbf{K}, \mathbf{L}) \leftarrow (h, h(\hat{\mathbf{k}}), h(\hat{\mathbf{l}})) \in \mathbf{R}^m \times \mathbf{R} \times \mathbf{R}$.

Signing: $\mathsf{Sign}((\hat{\mathbf{k}}, \hat{\mathbf{l}}), msg)$, on input a message $msg \in \mathbf{R}_0$ with $\|msg\|_\infty \leq 1$, outputs $\sigma \leftarrow \hat{\mathbf{k}} msg + \hat{\mathbf{l}} \in \mathbf{R}_0^m$.

Verification: $\mathsf{Vf}((h, \mathbf{K}, \mathbf{L}), \sigma, msg)$ returns 1 if and only if $\|\sigma\|_\infty \leq d$ and $h(\sigma) = \mathbf{K} msg + \mathbf{L}$.

Let $\mathbf{R}_b \subseteq \mathbf{R}_0$ denote the set of polynomials $\mathbf{p} \in \mathbf{R}_0$ with $\|\mathbf{p}\|_\infty \leq b$. If there is a successful adversary against unforgeability of LM-OTS, this adversary can be used to solve the collision problem $\text{COL}(\mathcal{H}(\mathbf{R}, m), \mathbf{R}_d)$.

For our masking scheme, we require a slightly looser version LM-OTS′ of LM-OTS, which allows for larger signatures in Vf. Key generation and signing remain unchanged and honestly created signatures are still in $\mathbf{R}_d^m$. The modified verification algorithm is defined as follows, where $\phi \in \mathbb{N}$ is a constant.

Verification: $\mathsf{Vf}((h, \mathbf{K}, \mathbf{L}), \sigma, msg)$ returns 1 if and only if $\|\sigma\|_\infty \leq 2\phi mnd - d$ and $h(\sigma) = \mathbf{K} msg + \mathbf{L}$.

Of course, we require a stronger assumption now. A successful adversary against LM-OTS′ can be used to solve $\text{COL}(\mathcal{H}(\mathbf{R}, m), \mathbf{R}_{2\phi mnd-d})$.

A Masking Scheme for LM-OTS $\text{MS}_{\text{LM-OTS}'}$ works as follows.

Sets: The masking values are chosen from $\mathcal{S} = \mathbf{R}_{\phi mnd}^m$ with the component-wise uniform distribution Ψ. The space of signatures is $\Sigma = \mathbf{R}_{2\phi mnd-d}^m$, the advice strings are in $\mathcal{V} = \mathbf{R}$, and masked signatures need to be in $\mathcal{T} = \mathbf{R}_{\phi mnd-d}^m$.

Advice: $\mathsf{Advice}((h, \mathbf{K}, \mathbf{L}), \alpha)$ outputs $\beta \leftarrow h(\alpha)$.

Mask: $\mathsf{Mask}((h, \mathbf{K}, \mathbf{L}), \sigma, \alpha, msg)$, on input $\sigma \in \mathbf{R}_d^m \subset \Sigma$, computes $\tau \leftarrow \sigma + \alpha$. If $\tau \in \mathcal{T}$, it returns τ and $\perp$ otherwise.

Unmask: $\mathsf{Unmask}(\tau, \beta, \alpha, msg)$ returns $\sigma \leftarrow \tau - \alpha$.

Verification: $\mathrm{MS}_{\mathrm{DS}}.\mathsf{Vf}((h, \mathbf{K}, \mathbf{L}), \tau, \beta, msg)$ outputs 1 if and only if $\tau \in \mathcal{T}$ and $h(\tau) = \mathbf{K}msg + \mathbf{L} + \beta$.

The above masking scheme yields correct masked signatures τ with constant probability $e^{-1/\phi}$ (cf. Lemma 4.1) if Mask is applied to honestly created signatures from $\mathbf{R}_d^m$. Notice that we explicitly allow such completeness defects in our construction; the algorithm Create simply starts over with a new α. Refer to Section 4.3 for a discussion of the parameter ϕ, which can be used to control the completeness defect. In total, we expect Mask to output a valid masked signature after $e^{1/\phi}$ trials, or asymptotically $g(n)$ for any $g(n) = \omega(\log(n))$.

The following propositions show that $\mathrm{MS}_{\text{LM-OTS}'}$ is indeed applicable, namely it is binding and hiding.

Proposition 5.13 (Binding)**.** *$MS_{LM\text{-}OTS'}$ is binding.*

Proof. Let $(h, \mathbf{K}, \mathbf{L})$ be any public key for LM-OTS′. Let τ be a masked signature for msg with advice $\beta = h(\alpha)$. If τ is valid under $\mathsf{Mask.Vf}$, then $h(\tau) = \mathbf{K}msg + \mathbf{L} + \beta$. Let $\sigma \leftarrow \tau - \alpha$ be the extracted signature. Then, obviously, $h(\sigma) = h(\tau) - h(\alpha) = \mathbf{K}msg + \mathbf{L}$. Furthermore, we always have $\|\sigma\|_\infty \leq \|\tau\|_\infty + \|\alpha\|_\infty \leq 2\phi mnd - d$. Thus, the extracted signature is always valid if τ is valid. □

Proposition 5.14 (Hiding)**.** *$MS_{LM\text{-}OTS'}$ is hiding if $COL(\mathcal{H}(\mathbf{R}, m), \mathbf{R}_{2\phi mnd-d})$ is hard in the presence of an LM-OTS′ signature oracle.*

Proof. Let $\mathcal{A}$ be an efficient, successful adversary in $\mathsf{Exp}^{\mathsf{UnMask}}_{\mathcal{A},\mathrm{MS}}$. We build a reduction $\mathcal{B}$ against the collision problem that has access to an LM-OTS′ signature oracle OSign for a public key $spk = (h, \mathbf{K}, \mathbf{L})$. Since LM-OTS′ is one-time, the adversary may *either* query Mask *or* OSign, but only once. Hence, in order to win the game, $\mathcal{A}$ has to query a message msg^* to Mask. The oracle is simulated honestly as follows. $\mathcal{B}$ queries msg^* to OSign and obtains σ. Then, it chooses $\alpha \leftarrow_\$ \mathcal{S}$ and sets $\beta \leftarrow h(\alpha)$. Finally, it computes $\tau \leftarrow \sigma + \alpha$ and returns τ if $\tau \in \mathcal{T}$ and $\perp$ otherwise. Eventually, $\mathcal{A}$ stops and outputs σ^*, such that $\text{LM-OTS}'.\mathsf{Vf}(spk, \sigma^*, msg^*) = 1$. We argue that $\sigma^* \neq \sigma$, which yields a solution to COL with norm bound $2\phi mnd - d$.

Notice that h admits collisions in Σ, i.e., for every $\sigma_1 \in \Sigma$ there is a second $\sigma_2 \in \Sigma \setminus \{\sigma_1\}$ with $h(\sigma_1) = h(\sigma_2)$. Obviously, $\mathcal{A}$ cannot learn anything about σ if $\tau = \perp$. Hence, with probability at least $1/2$, we have $\sigma^* \neq \sigma$ and obtain the required collision.

In the case that $\tau \neq \perp$, we apply a adaptation of Lemma 4.7 for views of the form $(\beta, 1, \tau)$ for $1 \in \mathbf{R}_0$. It establishes witness indistinguishability w.r.t. two colliding signatures (σ_1, σ_2). So, also with probability at least $1/2$, $\mathcal{A}$ outputs $\sigma^* \neq \sigma$. □

Using Proposition 2.3 we can base the hiding property on the worst-case hardness of SIVP. The condition that the underlying problem needs to be hard in the presence of an LM-OTS′ signature oracle can be lifted by essentially reproving the security of LM-OTS′ in analogy to [LM08].

Corollary 5.15. *$MS_{\text{LM-OTS}'}$ is hiding if, in the presence of an LM-OTS′ signature oracle, solving SIVP$^{\infty}$ is hard in the worst case for approximation factors $\gamma = \widetilde{\mathcal{O}}(n^3)$ in lattices that correspond to ideals in $\mathbf{R}_0$.*

5.5 Conclusion and Open Problems

With our work, we have removed a loophole in the model of Boneh et al. by introducing extractability and non-frameability as novel security requirements for verifiably encrypted signature schemes.

Furthermore, we have extended the specification for such schemes with an optional setup phase, which we believe is common in real-world scenarios. Also we have proposed to allow for a limited signature capacity to support the following, interesting business model. The adjudicator charges its clients on a per signature-exchange basis. Hence, they need to acquire exchange tokens and spend them one at a time. Once all of them have been spent, the user simply request a fresh set of tokens. There is no need to generate a new signature key.

Moreover, our proposed constructions support signature laws that allow for legally binding contracts via digital signatures. In particular, we have shown the first pairing-free VES scheme in the standard model. Instead, it can be based entirely on hard lattice problems.

An interesting further research question is to combine previous efficiency improvements for Merkle-type signature schemes with our approach to increase the signature capacity. Also, the computational bottleneck for our scheme seems to be the employed encryption scheme, which has to have a large throughput for the setup phase to be efficient. It would be interesting to see more research on high-performance encryption schemes from lattices.

Finally, an obvious way to strengthen our security model is to remove the trusted setup phase, where we assume the adjudicator to be honest. Ensuring extractability and non-frameability under dishonest adjudicators seems to be very hard to achieve.

Chapter 6

Single-signer Aggregate Signatures

When compared to classical signature schemes based on, e.g., the discrete logarithm problem, lattice-based signature are still rather large. Especially when transmitting vast amounts of signatures over a network or storing them in a long-term archive, large signature sizes may be prohibitive.

We put forward the notion of single-signer aggregate signature (SSA) schemes along with a formal security model. Our definition extends the specification of ordinary signature schemes with a public aggregation algorithm. The algorithm takes as input a list of signatures under the *same* public key and outputs an aggregate signature. Given the aggregate and the list of corresponding messages, the verification procedure validates the aggregate. An interesting extension of our model is *hierarchical aggregation*, i.e., the aggregation of aggregates.

Note that the aggregation algorithm does not have access to any secret key. Otherwise, a trivial aggregation strategy would use the secret key to sign the entire list of messages again to output a single one. Another trivial aggregate would be to simply append all signatures as bit strings but obviously, we are interested in compressing the input as much as possible. In our opinion, an SSA scheme is feasible as soon as the size of the aggregate grows sub-linearly with the number of aggregated signatures. However, we do not make this a strict requirement as even smaller savings may be worthwhile.

Our proposal can be interpreted as a restricted form of aggregate signature scheme [BGLS03]. There, the goal is to also aggregate signatures of different signers, whereas we demand the signatures to be valid under the same public key. Multi signatures [BN06] provide another restricted form of aggregation, where multiple signers are allowed but the signed messages need to be equal. Hence, multi signatures and single-signer aggregates are dual primitives. Another related line of work is batch verification [BGR98a, BGR98b, CHP07], with the purpose of improving the efficiency of multi-signature verification. Data compression is not a requirement here. When compressing signatures, however, the verification cost is likely to decrease as well.

To date, full-blown aggregate signature schemes can only be built upon pairings. In addition, we believe that they are often not required and discussing restricted forms thereof may be more suitable for real-world applications. Two immediate applications of SSA spring to mind, one for the receiver and one for the sender of signatures.

CRL Compression Signed certificate revocation lists (CRLs) are issued in public-key infrastructures (PKI) to manage the premature revocation of signatures. Let us

consider the case of frequently updated "delta" CRLs [CSF+08]. A client downloads a full CRL and daily updates in the form of delta CRLs until a new "full" CRL is available. The signed data in a delta CRL is typically small, basically consisting of a small number of certificate identifiers. Now to protect against local malware, the client stores the list of revoked keys *and* the signature. When following Microsoft's recommendation for CRL refresh intervals [Mic10], we are faced with storing around 90 to 180 signatures before receiving a new full CRL, which may be a problem on resource-constrained devices. Using our primitive, the amount of signatures to be kept is constant and verification is potentially more efficient as well.

Delayed Signature Transmission On the sender's end, using SSA may also be beneficial. A topical example is the emerging use of cryptography for medical data. Take your family doctor for example. He or she digitally signs dozens of prescriptions per day. For archiving and accounting reasons, the signatures are kept in his or her practice and they are to be sent to a health insurance proxy for accounting. To save both bandwidth and storage, the doctor's receptionist may aggregate the signatures on a regular basis and transmit the aggregate to the accounting proxy. Thinking one step further, the receptionist may even apply our idea of hierarchical aggregation and aggregate, e.g., all daily aggregates on a weekly basis.

Our Contribution We introduce single-signer aggregate signatures as a new primitive and equip it with two security notions akin to existential and strong unforgeability. Then, we propose a construction SSA, based on the lattice-based PSTF in Section 2.7.3 and prove the following theorem.

Theorem (Aggregate Unforgeability)**.** *SSA, aggregating up to ℓ_{max} signatures, is unforgeable if SIS is hard for norm bounds $\nu = \ell_{max}\widetilde{\mathcal{O}}(n)$.*

Hence, our scheme can aggregate up to polynomially many signatures. Verifying an aggregate only requires one matrix-vector product, regardless of the number of aggregated signatures. More importantly, the aggregate only grows logarithmically with the number of aggregated signatures. Since the assumption depends on the maximum number of aggregated signatures, the main security parameter needs to grow with it as well. To compensate for that, we analyze the net advantage of our scheme based on the hardness estimates in Chapter 3 and find that the saving are still considerable.

Organization We provide a formal specification and two security models (existential and strong) in Section 6.1. Then, we propose a simple lattice-based instantiation in Section 6.2, propose secure parameters, and conclude the chapter in Section 6.3.

This chapter contains an excerpt from [CNR10], of which the dissertation author is the primary investigator and author. There, SSA schemes are used to implement anonymous attribute tokens — a form of anonymous credentials with multiple attributes and selective attribute revealing [Bra99, CL01].

6.1 Specification and Security

The specification of $\mathrm{SSA} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Agg}, \mathsf{Vf})$ extends the one for signature schemes $\mathrm{DS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ with an algorithm Agg. On input a list of signatures under the same public key, the algorithm outputs an aggregate signature Σ. The verification algorithm takes as input the public key, a list of messages, and the aggregate. It outputs 1 if and only if the aggregate is valid w.r.t. the message list. More formally:

Key Generation: $\mathsf{Kg}(1^n)$ outputs a secret key sk and a public key pk.

Signing: $\mathsf{Sign}(sk, msg)$, on input a message $msg \in \mathcal{M}$ from the message space $\mathcal{M}$, outputs a signature σ for msg.

Aggregation: $\mathsf{Agg}((\sigma_i)_{i=1}^{\ell})$, on input a list of signatures, outputs an aggregate signature Σ. For $\ell = 1$, it returns $\Sigma \leftarrow \sigma_1$.

Verification: $\mathsf{Vf}(pk, \Sigma, (msg_i)_{i=1}^{\ell})$ outputs 1 if and only if Σ is a valid aggregate signature for the list $(msg_i)_{i=1}^{\ell}$, $\ell \leq \ell_{max}$, under pk and regardless of the order of messages.

Correctness is straightforward, i.e., for all honestly generated keys, signatures, and aggregates, Vf outputs 1 with overwhelming probability. Notice that Vf can also be queried with $\ell = 1$ to verify ordinary signatures of the same type.

We model security in the existential and in the strong sense. In both models, the adversary $\mathcal{A}$ has adaptive access to a signature oracle. Let $((msg_i, \sigma_i))_{i=1}^{Q_{\mathsf{OSign}}}$ be the corresponding message-signature signature pairs. In order for a scheme to satisfy existential aggregate-unforgeability (SSA-EU-CMA), $\mathcal{A}$ in the experiment $\mathsf{Exp}^{\mathsf{SSA\text{-}EU\text{-}CMA}}_{\mathcal{A},\mathrm{SSA}}$ must not be able to produce a *valid* aggregate Σ^* for *distinct* messages $(msg_i^*)_{i=1}^{\ell^*}$, $\ell^* \leq \ell_{max}$, such that $\{msg_i^*\}_{i=1}^{\ell^*}$ contains a message that has not been queried to OSign before. Demanding distinct messages makes sense in practice and simplifies the following formal security requirements.

Experiment $\mathsf{Exp}^{\mathsf{SSA\text{-}EU\text{-}CMA}}_{\mathcal{A},\mathrm{SSA}}(n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$((msg_i^*)_{i=1}^{\ell^*}, \Sigma^*) \leftarrow \mathcal{A}^{\mathsf{OSign}(sk,\cdot)}(pk)$
Let $((msg_i, \sigma_i))_{i=1}^{Q_{\mathsf{OSign}}}$ be the query-answer pairs of $\mathsf{OSign}(sk, \cdot)$.
Return 1 iff $\ell^* \leq \ell_{max}$
and $\mathsf{Vf}(pk, \Sigma^*, (msg_i^*)_{i=1}^{\ell^*}) = 1$
and $\{msg_i^*\}_{i=1}^{\ell^*} \setminus \{msg_i\}_{i=1}^{Q_{\mathsf{OSign}}} \neq \emptyset$.

The scheme satisfies strong aggregate-unforgeability (SSA-SU-CMA), if no $\mathcal{A}$ is able to output a non-trivial, valid aggregate Σ^*, i.e., it does not output a straightforward aggregate of any subset of the received signatures. Whether or not $\mathcal{A}$ uses a new message in its forgery is irrelevant. We formalize this as follows.

Experiment $\mathsf{Exp}^{\mathsf{SSA\text{-}SU\text{-}CMA}}_{\mathcal{A},\mathrm{SSA}}(n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$((msg_i^*)_{i=1}^{\ell^*}, \Sigma^*) \leftarrow \mathcal{A}^{\mathsf{OSign}(sk,\cdot)}(pk)$
Let $((msg_i, \sigma_i))_{i=1}^{Q_{\mathsf{OSign}}}$ be the query-answer pairs of $\mathsf{OSign}(sk, \cdot)$.
Return 1 iff $\ell^* \leq \ell_{max}$
and $\mathsf{Vf}(pk, \Sigma^*, (msg_i^*)_{i=1}^{\ell^*}) = 1$
and $\forall R \subseteq [Q_{\mathsf{OSign}}] : \Sigma^* \neq \mathsf{Agg}((\sigma_i)_{i \in R})$.

As usual, an adversary is successful in either model if it is efficient and the respective experiment outputs 1 with non-negligible probability. Notice that SSA-SU-CMA requires Agg to be deterministic.

Hierarchical Aggregation A potentially interesting extension of our model is to support hierarchical aggregation, i.e., the aggregation of aggregate signatures. Here, we let Agg operate on a list of aggregates $(\Sigma)_{i=1}^{\ell}$ instead of on signatures, with signatures $\Sigma = (\sigma_i)_{i=1}^{1}$ being a special case. The output would be another aggregate Σ and the upper bound ℓ_{max} is already implicitly enforced in Vf.

Hard Aggregate Extraction Another extension would be to demand the hardness of inverting the Agg procedure, i.e., to extract or remove individual signatures from a (non-trivial) aggregate.

6.2 Our Construction

Our scheme SSA = $(\mathsf{Kg}, \mathsf{Sign}, \mathsf{Agg}, \mathsf{Vf})$ builds upon the GPV scheme [GPV08]. Hence, it uses the family PSTF = $(\mathsf{Kg}, \mathsf{Eval}, \mathsf{SampleDom}, \mathsf{SamplePre})$ from Section 2.7.3 and a full-domain hash random oracle $\mathsf{H} : \{0,1\}^* \to \mathbb{Z}_q^n$. It can aggregate up to $\ell_{max} \leq \mathrm{poly}(n)$ signatures.

Key Generation: $\mathsf{Kg}(1^n)$ runs $(\mathbf{A}, \mathbf{T}) \leftarrow \mathrm{PSTF.Kg}(1^n)$ and outputs $(sk, pk) \leftarrow (\mathbf{T}, \mathbf{A})$.

Signing: $\mathsf{Sign}(\mathbf{T}, msg)$, on input a message $msg \in \{0,1\}^*$, computes $\mathbf{h} \leftarrow \mathsf{H}(msg)$, samples $\sigma \leftarrow \mathsf{SamplePre}(\mathbf{T}, \eta, \mathbf{h})$, and returns σ. The algorithm is stateful, i.e., repeated queries with msg are answered with the same σ. [1]

Aggregation: $\mathsf{Agg}((\sigma_i)_{i=1}^{\ell})$ outputs $\Sigma \leftarrow \sum_{i=1}^{\ell} \sigma_i$.

Verification: $\mathsf{Vf}(\mathbf{A}, \Sigma, (msg_i)_{i=1}^{\ell})$ verifies that $0 < \|\Sigma\|_2 \leq \ell\eta\sqrt{m}$, all messages are distinct, and $\mathbf{A}\Sigma \equiv \sum_{i=1}^{\ell} \mathsf{H}(msg_i)$. It outputs 1 if and only if all conditions are satisfied.

Due to the linearity of the $\mathbf{A}$, the scheme is correct. Let $d^\infty := \eta\theta(m)$ be the upper bound for the infinity norm of a single signature. Then, an ℓ-aggregate requires only $m \log_2(\ell d^\infty)$ bits of storage, instead of $m\ell \log_2(d^\infty)$ for a trivial aggregate. See Figure 6.1 for a comparison of our aggregate signature scheme against storing GPV signatures individually. The underlying parameters have been chosen according to Chapter 3 so that they are secure against the attacker "Lenstra" until 2018. The main parameter is $n = 330$ and a single signature occupies about 154 kilobytes (kB). Moreover, note that instead of ℓ matrix-vector products over $\mathbb{Z}_q$, our verification algorithm only requires a single one. This constitutes a significant improvement, also in the sense of a batch verification scheme. Also, the overall efficiency can be improved via ideal lattices.

The potential savings are intriguing, but they come at the price of an assumption that depends on ℓ_{max}.

Theorem 6.1. *SSA is SSA-SU-CMA secure in the random oracle model if SIS*$(n, m, q, 2\ell_{max}\eta\sqrt{m})$ *is hard.*

[1] It is easy to make the scheme stateless by including a random string r in the hash, so that $msg\|r$ is unique with overwhelming probability. The random strings r are not aggregated but the size of σ dominates the signature size. We omit this modification.

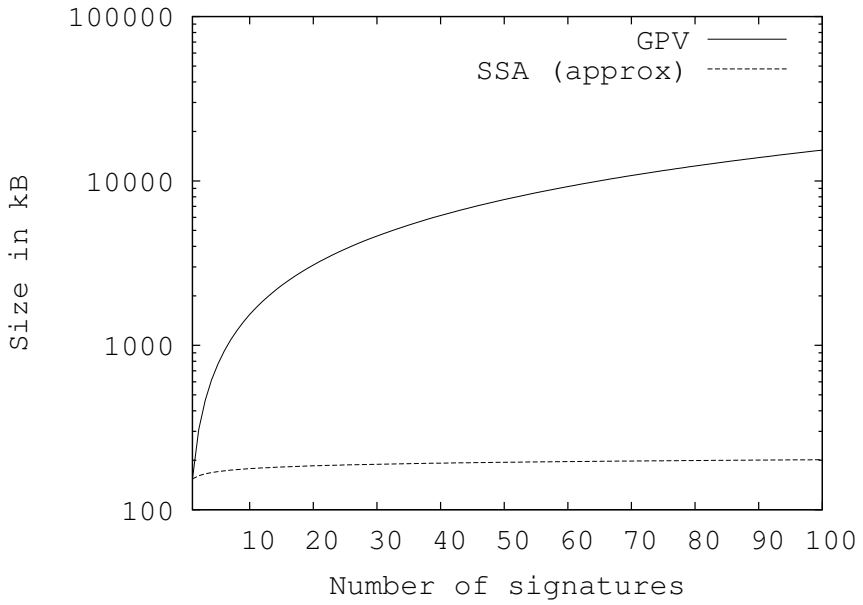


Figure 6.1: Comparison of the storage requirements for SSA signatures against individual storage of GPV [GPV08] signatures. The y-axis is in logarithmic scale.

Proof. Let us assume an efficient, successful adversary $\mathcal{A}$ that breaks strong aggregate-unforgeability of SSA. We build a reduction $\mathcal{B}^{\mathcal{A}}$ to solve SIS as follows.

Setup: The reduction receives $\mathbf{A}$ from the SIS problem and forwards it to $\mathcal{A}$.

Simulation: $\mathcal{B}$ maintains a list $ListSIG$ of message-hash-signature triples to answer all queries to Sign and H consistently. Upon a *new* hash query msg_i, $\mathcal{B}$ chooses $\sigma_i \leftarrow \mathsf{SampleDom}(m, \eta)$ and programs $\mathsf{H}(msg_i) := \mathbf{A}\sigma_i \bmod q$.

Output: Eventually, $\mathcal{A}$ outputs a forgery Σ^* for $(msg_i^*)_{i=1}^{\ell^*}$. The reduction looks up signatures σ_i^* for msg_i^* in $ListSIG$. If there is not such entry, it is generated in-place by calling $\mathsf{H}(msg_i^*)$. The final output is $\mathbf{x} \leftarrow \Sigma^* - \sum_{i=1}^{\ell^*} \sigma_i^*$.

Analysis Observe that the hash and signature oracles are simulated correctly and efficiently due to the properties of PSTF (cf. Section 2.7.3).

Since we use the random oracle to simulate all queries, we can safely assume that $\mathcal{A}$ has made a query $\mathsf{H}(msg_i^*)$ for all $i \in [\ell^*]$ and, therefore, the reduction has a signature σ_i^* such that $\mathbf{A}\sigma_i^* \equiv \mathsf{H}(msg_i^*)$ in *ListSIG*. $\mathcal{A}$'s forgery is legitimate if

1. $0 < \|\Sigma^*\|_2 \leq \ell^* \eta \sqrt{m}$;
2. For all subsets $R \subseteq [Q_{\mathsf{OSign}}]$ we have $\Sigma^* \neq \sum_{i \in R} \sigma_i^*$;
3. $\mathbf{A}\Sigma^* \equiv \sum_{i=1}^{\ell^*} \mathsf{H}(msg_i^*)$.

Via (1), the reduction's output $\mathbf{x}$ has length $\|\mathbf{x}\|_2 \leq \left\|\Sigma^* - \sum_{i=1}^{\ell^*} \sigma_i^*\right\|_2 \leq 2\ell^* \eta \sqrt{m}$. Condition (2) and the high conditional min-entropy of signatures imply $\Sigma^* \neq \sum_{i=1}^{\ell^*} \sigma_i^*$ and therefore $\mathbf{x} \neq \mathbf{0}$. Finally, (3) establishes $\mathbf{x} \in \Lambda_q^{\perp}(\mathbf{A})$. □

It is possible to prove a slightly tighter theorem for existential unforgeability that requires SIS to be hard for $\nu = (2\ell_{max} - 1)\eta\sqrt{m}$ instead. Via Proposition 2.1, we arrive at a reduction from worst-case lattices problems.

Corollary 6.2. *SSA is SSA-SU-CMA secure if SIVP is hard in the worst case for approximation factors* $\gamma = \widetilde{\mathcal{O}}(\ell_{max} n \sqrt{n})$.

Secure Parameters In the above discussion and in Figure 6.1, we have seen the approximate advantage of our aggregate signature scheme over storing individual signatures. We have also seen that security degrades with increasing ℓ_{max}, which renders the observed advantage unfair. In the following, we use our framework in Chapter 3, fix a security level — security until 2018 against the attacker "Lenstra" — and compute parameter sets for $1 \leq \ell_{max} \leq 10000$ with the same security level. Let q be as required for single signatures ($\ell_{max} = 1$) according to Proposition 2.1. Then, for a given ℓ_{max}, we take the modulus $q' \leftarrow \ell_{max} q$, update the dependent parameter relations, and increase n until the desired security level is attained.

The result is depicted in Figure 6.2, which shows the net advantage of our scheme with a steady security level.

6.3 Conclusion and Open Problems

We have introduced a new aggregate signature primitive for the special case of a single signer, where the aggregation algorithm can be run by anyone without knowing the secret key. Furthermore, we have demonstrated that such a scheme can

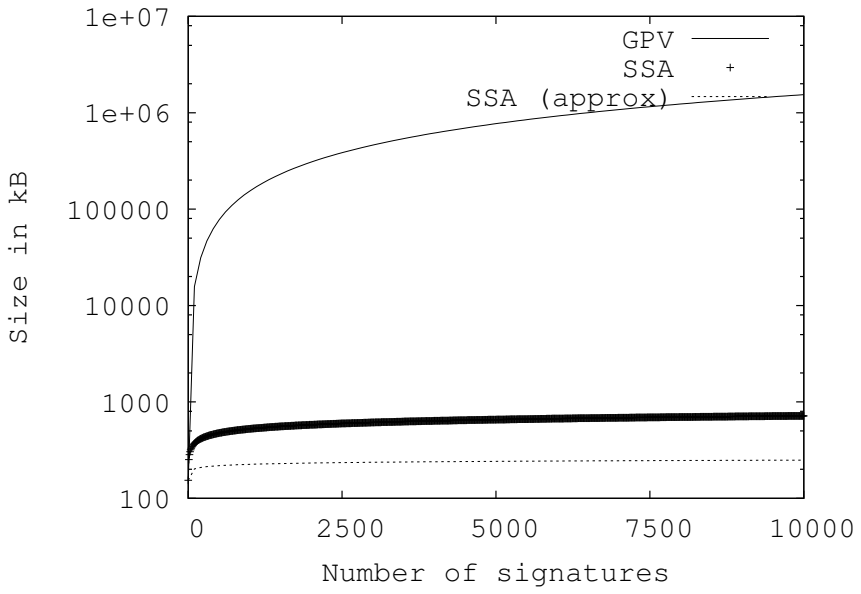


Figure 6.2: Net signature size in SSA, after establishing conjectured security until 2018 against the attacker "Lenstra".

be easily built from standard tools in lattice cryptography and that the resulting computational as well as bandwidth savings are striking.

Achieving a full-fledged aggregate or multi signature scheme from lattices remains a challenging open research problem,

Chapter 7

Strongly Unforgeable Signatures in the Standard Model

Basically, there are two classes of signature schemes, tree-based and treeless. *Tree-based* Merkle signature schemes [Mer89] can be built upon hash functions and one-time signature schemes. They have a limited signature capacity and their efficiency depends on this capacity. See [BDS08] for an overview. *Treeless* constructions are typically more efficient, allow for an unlimited number of signatures, and are easier to handle in practice. Therefore, in practice, we almost exclusively use schemes that fall into the second category.

Based on lattices, we have seen numerous signature schemes appear recently. In the random oracle model there are schemes due to Gentry, Peikert, and Vaikuntanathan [GPV08]; Stehlé, Steinfeld, Tanaka, and Xagawa [SSTX09]; and Lyubashevsky [Lyu09]. As for the standard model, there are the works of Lyubashevsky and Micciancio [LM08] (one-time); Cash, Hofheinz, Kiltz, and Peikert [CHKP10]; and Boyen [Boy10].

However, there is a gap in this range because none of the above schemes is *treeless*, provably secure in the *standard model*, and *strongly unforgeable*. Recall that, in EU-CMA, the adversary is forced to output a signature for a *fresh message* msg^* after seeing signatures for messages $msg_i \neq msg^*$ of his or her choice. The SU-CMA adversary is also allowed to output a *fresh signature* for one of the previously queried msg_i (cf. Section 2.3.4).

Strong unforgeability is interesting in both, theory and practice. Consider signcryption [ADR02] or generic transformations from CPA to chosen ciphertext (CCA) security in the standard model, e.g., Dolev, Dwork, and Naor [DDN00] or Boneh, Canetti, Halevi, and Katz [BCHK07]. They typically involve a strongly unforgeable signature scheme to make the ciphertext authentic and non-malleable.

As a simple practical example, let us consider an access control protocol where you may delegate certain rights to another party by signing a description for these rights with a signature σ. You want to be able to revoke them at any time in the future via an online revocation system. The rights are revoked as soon as the online system has σ in its database. If the signature scheme is only EU-CMA secure, the delegate can construct another signature σ^* for the same set of rights and present this token instead of σ — the revocation mechanism breaks down. This naive example demonstrates that great care must be taken when designing protocols based on EU-CMA.

There is a generic transformations from EU-CMA to SU-CMA due to Boneh, Shen, and Waters [BSW06]. It only works for a certain subclass of signature schemes, which does not contain any known lattice-based construction. More generic transformations, such as Bellare and Shoup's work [BS07], require an additional signature scheme and harm efficiency.

Our Contribution We build upon the EU-CMA signature scheme in [CHKP10] and construct an SU-CMA scheme. Both constructions share the same "hash-sign-switch" approach of constructing a signature scheme in an even weaker *static message attack* (SMA) model before applying a generic transformation [KR00] to obtain *chosen message attack* (CMA) security. Our scheme also yields a strongly-unforgeable online/offline signature scheme [EGM96, ST01].

In the SMA model, the adversary is forced to submit all signature oracle queries before seeing the public key. In a proof, this relaxation often helps the reduction to prepare for the simulation of the oracle by "rigging" the public key accordingly. It is well-known that existential unforgeability under static message attacks (EU-SMA) implies EU-CMA security if chameleon hash functions exist. We extend this result to a conversion from **SU**-SMA to **SU**-CMA with the following theorem.

Theorem (Generic Conversion)**.** *SU-SMA implies SU-CMA if chameleon hash functions exist.*

Then, we construct a lattice-based signature scheme DS that is SU-SMA secure in the standard model and apply the above theorem. Our construction is essentially as efficient as [CHKP10]. In our case, signing involves a simple additional linear algebra step that can be pre-computed during key generation. Hence, we achieve a stronger security notion without additional cost and prove the following theorem.

Theorem (Main Construction)**.** *DS is SU-CMA secure in the standard model if chameleon hash functions with range* $\{0,1\}^\lambda$ *exist and SIS is hard with norm bounds* $\nu = \widetilde{\mathcal{O}}(n\sqrt{\lambda})$*.*

Our Modifications to [CHKP10] For those familiar with [CHKP10], we give a brief overview of the changes that are necessary to achieve SU-CMA security. A similar trick works for Boyen's more recent scheme [Boy10] as well.

In [CHKP10], signatures are short vectors σ that satisfy $\mathbf{A}_{msg}\sigma \equiv \mathbf{0} \pmod q$, i.e., they are in the q-ary lattice $\Lambda_q^\perp(\mathbf{A}_{msg})$ with a public $\mathbf{A}_{msg}$ depending on the message. An adversary may succeed in breaking SU-CMA security of this scheme by simply asking for a signature σ for a message msg and then return $(msg, -\sigma)$ as its forgery. Such an answer is useless in the reduction because the simulator had to know a trapdoor for $\Lambda_q^\perp(\mathbf{A}_{msg})$ beforehand to generate σ.

Instead, we let the signature algorithm sample short vectors from a random coset $\{\mathbf{x} : \mathbf{A}_{msg}\mathbf{x} \equiv \mathbf{y} \pmod q\}$ of $\Lambda_q^\perp(\mathbf{A}_{msg})$. The approach is similar to [GPV08] but with a fixed $\mathbf{y}$ that is part of the public key and without random oracles. In the simulation, we can prepare $\mathbf{y}$ such that we know a corresponding signature σ that is

used to simulate the signature oracle for msg. There is no need for the reduction to know a trapdoor for $\Lambda_q^{\perp}(\mathbf{A}_{msg})$. Now, the adversary against SU-CMA security needs to output a different short vector σ^* from the same coset. This, however enables the simulation to find a short vector $\sigma - \sigma^*$ in $\Lambda_q^{\perp}(\mathbf{A}_{msg})$ and solve the underlying problem.

Organization We discuss the relaxed security notion SU-SMA in Section 7.1 and explain how the transformation to SU-CMA works. Then, we instantiate the SU-SMA model from lattices in Section 7.2 as described above. Finally, we conclude the chapter in Section 7.3.

This chapter repeats the essential results in [Rüc10c]. The dissertation author was the primary investigator and author of this paper. The second result in the paper, an identity-based signature scheme, is effectively superseded by the construction principle in Chapter 8.

7.1 From Static to Chosen Message Security

Our main construction relies on a generic conversion from the weaker notion of strong unforgeability under static message attacks to the standard notion (under chosen message attacks) as defined in $\mathsf{Exp}^{\mathsf{SU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}$ in Section 2.3.4. After a brief definition of the security model, we describe our conversion with chameleon hash functions.

7.1.1 Security Model

We recap an analogous notion for existential unforgeability, namely EU-SMA, as introduced in [KR00]. The main difference to EU-CMA is that the adversary submits all messages $msg_1, \ldots, msg_{Q_{\mathsf{OSign}}}$ before seeing the public key and the corresponding signatures. It is defined in $\mathsf{Exp}^{\mathsf{EU\text{-}SMA}}_{\mathcal{A},\mathrm{DS}}$ for a signature scheme $\mathrm{DS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$, where $\mathcal{A}$ works in two modes and is allowed to keep a state.

Experiment $\mathsf{Exp}^{\mathsf{EU\text{-}SMA}}_{\mathcal{A},\mathrm{DS}}(n)$
$((msg_i)_{i=1}^{Q_{\mathsf{OSign}}}, s\_find) \leftarrow \mathcal{A}(m\_find, 1^n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$\sigma_i \leftarrow \mathsf{Sign}(sk, msg_i)$ for $i \in [Q_{\mathsf{OSign}}]$
$(msg^*, \sigma^*) \leftarrow \mathcal{A}(m\_forge, s\_find, pk, (\sigma_i)_{i=1}^{Q_{\mathsf{OSign}}})$
Return 1 iff $\mathsf{Vf}(pk, \sigma^*, msg^*) = 1$ and $msg^* \notin (msg_i)_{i=1}^{Q_{\mathsf{OSign}}}$.

Since we are interested in *strong* unforgeability, we modify $\mathsf{Exp}^{\mathsf{EU\text{-}SMA}}_{\mathcal{A},\mathrm{DS}}$ to allow for strong forgeries as in $\mathsf{Exp}^{\mathsf{SU\text{-}CMA}}_{\mathcal{A},\mathrm{DS}}$ as follows. The only difference is in the definition of a legitimate forgery.

Experiment $\mathsf{Exp}^{\mathsf{SU\text{-}SMA}}_{\mathcal{A},\mathrm{DS}}(n)$
$((msg_i)_{i=1}^{Q_{\mathsf{OSign}}}, s\_find) \leftarrow \mathcal{A}(m\_find, 1^n)$
$(sk, pk) \leftarrow \mathsf{Kg}(1^n)$
$\sigma_i \leftarrow \mathsf{Sign}(sk, msg_i)$ for $i \in [Q_{\mathsf{OSign}}]$
$(msg^*, \sigma^*) \leftarrow \mathcal{A}(m\_forge, s\_find, pk, (\sigma_i)_{i=1}^{Q_{\mathsf{OSign}}})$
Return 1 iff $\mathsf{Vf}(pk, \sigma^*, msg^*) = 1$ and $(msg^*, \sigma^*) \notin ((msg_i, \sigma_i))_{i=1}^{Q_{\mathsf{OSign}}}$.

Security in either model is defined as with EU-CMA and SU-CMA, respectively.

7.1.2 Transformation

A helpful fact about chameleon hash functions is that if they exist, then there is a generic transformation from EU-SMA to EU-CMA signatures. This was known since [KR00] and it is proven in [HW09]. We show that the *same* transformation also transforms *SU*-SMA into *SU*-CMA.

Theorem 7.1. *SU-SMA implies SU-CMA if chameleon hash functions exist.*

Proof. Let $\mathcal{C} = (\mathsf{Kg}, \Psi)$ be a chameleon hash function family (cf. Section 2.3.3) that maps into $\mathcal{M}$ and take any SU-SMA secure signature scheme $\mathrm{DS}^{\mathsf{SU\text{-}SMA}} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ with message space $\mathcal{M}$. Then, we can construct an SU-CMA secure scheme $\mathrm{DS}^{\mathsf{SU\text{-}CMA}} = (\mathsf{Kg}', \mathsf{Sign}', \mathsf{Vf}')$ as follows.

Key Generation: $\mathsf{Kg}'(1^n)$ runs $(sk, pk) \leftarrow \mathrm{DS}^{\mathsf{SU\text{-}SMA}}.\mathsf{Kg}(1^n)$ and $(\mathsf{C}, \mathsf{C}^{-1}) \leftarrow \mathcal{C}.\mathsf{Kg}(1^n)$. It outputs the secret key sk and the public key (pk, C).

Signing: $\mathsf{Sign}'(sk, msg)$ picks a random $\rho \leftarrow_\Psi \mathcal{R}$, computes $\mu \leftarrow \mathsf{C}(msg, \rho)$, and $\sigma \leftarrow \mathrm{DS}^{\mathsf{SU\text{-}SMA}}.\mathsf{Sign}(sk, \mu)$. The resulting signature is (σ, ρ).

Verification: $\mathsf{Vf}'((pk, \mathsf{C}), (\sigma, \rho), msg)$ computes $\mu \leftarrow \mathsf{C}(msg, \rho)$ and returns the result of $\mathrm{DS}^{\mathsf{SU\text{-}SMA}}.\mathsf{Vf}(pk, \sigma, \mu)$.

Towards contradiction, assume that there is an efficient and successful adversary $\mathcal{A}$ against SU-CMA security of $\mathrm{DS}^{\mathsf{SU\text{-}CMA}}$. Assume that the adversary queries the messages $(msg_i)_{i=1}^{Q_{\mathsf{OSign}}}$ to its signature oracle and receives $((\sigma_i, \rho_i))_{i=1}^{Q_{\mathsf{OSign}}}$. Now, a *successful* adversary can be classified into one of three types, depending on its output $(msg^*, \sigma^*, \rho^*)$.

1. $\exists i \in [Q_{\mathsf{OSign}}] : \mathsf{C}(msg^*, \rho^*) = \mathsf{C}(msg_i, \rho_i)$
   a) $\rho^* \neq \rho_i$ or $msg^* \neq msg_i$: we have a collision under C.
   b) $\rho^* = \rho_i$ and $msg^* = msg_i$: we have $\sigma^* \neq \sigma_i$, an SU-SMA forgery.
2. $\forall i \in [Q_{\mathsf{OSign}}] : \mathsf{C}(msg^*, \rho^*) \neq \mathsf{C}(msg_i, \rho_i)$: we have an SU-SMA (even EU-SMA) forgery.

Type-1a adversaries find collisions under C, therefore the reduction has to play against collision resistance of the family $\mathcal{C}$. Type-1b adversaries find a forgery in the strong sense, i.e., the reduction plays against SU-SMA security of $\mathrm{DS}^{\mathsf{SU\text{-}SMA}}$. Type-2 adversaries $\mathcal{A}$ always output an existential forgery that refutes SU-SMA, and even EU-SMA, security of $\mathrm{DS}^{\mathsf{SU\text{-}SMA}}$. Whenever we expect the adversary to be of type 1a, we simulate the environment with the secret signing key sk. Otherwise, we receive the public verification key from the SU-SMA experiment and simulate the signature oracle with the trapdoor C^{-1} for the chameleon hash. However, the adversary's view in both reductions is indistinguishable. We describe both reductions.

Type-1a We describe a reduction that refutes collision resistance of the chameleon hash function family $\mathcal{C}$. The reduction receives a random function from the family $\mathcal{C}$.

Setup: Receive C and run $(sk, pk) \leftarrow \mathrm{DS}^{\mathsf{SU\text{-}SMA}}.\mathsf{Kg}(1^n)$. Then, run $\mathcal{A}$ on input (pk, C).

Signature Queries: On input msg, choose a random $\rho \leftarrow_{\$} \{0,1\}^n$. Then, compute $\mu \leftarrow \mathsf{C}(msg, \rho)$ and return the result of $\mathrm{DS}^{\mathsf{SU\text{-}SMA}}.\mathsf{Sign}(sk, \mu)$ and ρ.

Output: When $\mathcal{A}$ outputs $(msg^*, \sigma^*, \rho^*)$, the reduction outputs the collision (msg, μ), (msg^*, μ^*).

Analysis Observe that the environment of $\mathcal{A}$ is perfectly simulated. By definition, a type-1a forger outputs a signature (σ^*, ρ^*) and a message msg^* such that $\mathsf{C}(msg^*, \rho^*) = \mathsf{C}(msg_i, \rho_i)$ for some i but with $msg^* \neq msg_i$ or $\rho^* \neq \rho_i$. Therefore, the output is a valid collision under C and the reduction is successful whenever $\mathcal{A}$ is. The overhead of the reduction is negligible.

Type-1b/2 We describe a reduction that refutes SU-SMA security of $\mathrm{DS}^{\mathsf{SU\text{-}SMA}}$. The reduction has access to an external signature oracle in the setup phase.

Setup: Choose Q_{OSign} uniformly random messages $(\mu_i)_{i=1}^{Q_{\mathsf{OSign}}}$ and send them to the signature oracle. Receive $(\sigma_i)_{i=1}^{Q_{\mathsf{OSign}}}$ and pk. Choose $(\mathsf{C}, \mathsf{C}^{-1}) \leftarrow \mathcal{C}.\mathsf{Kg}(1^n)$ and execute $\mathcal{A}$ on input (pk, C). Set up a counter $\imath \leftarrow 0$.

Signature Queries: On input msg, increment $\imath$ and compute $\rho_i \leftarrow \mathsf{C}^{-1}(\mu_i, msg)$. The oracle returns (μ_i, σ_i).

Output: Eventually, the adversary $\mathcal{A}$ outputs $(msg^*, \sigma^*, \rho^*)$. Then, the reduction returns $(\mathsf{C}(msg^*, \rho^*), \sigma^*)$.

Analysis Observe that the environment of $\mathcal{A}$ is perfectly simulated. By definition, a type-1b forger outputs a signature (σ^*, ρ^*) and a message msg^* such that $\mathsf{C}(msg^*, \rho^*) = \mu_i$ for some $i \in [Q_{\mathsf{OSign}}]$ and $msg^* = msg_i$ as well as $\rho^* = \rho_i$. Since it is a forgery, we have that $\sigma^* \neq \sigma_i$. Therefore, σ^* is a forgery for μ_i in the strong sense.

A type-2 forger outputs a pair (msg^*, ρ^*) such that $\mathsf{C}(msg^*, \rho^*) \neq \mu_i$ for all $i \in [Q_{\mathsf{OSign}}]$. Thus, the reduction outputs a forgery in the existential sense because $\mathsf{C}(msg^*, \rho^*)$ has never been queried to the SU-SMA signature oracle. In either case, the reduction is successful whenever $\mathcal{A}$ is. The overhead is dominated by the calls to C^{-1}. □

7.2 An Instantiation with Lattices

In the following, we propose our main construction, an SU-SMA secure signature scheme in the standard model based on the SIS problem. When combined with Theorem 7.1, we obtain an SIS-based SU-CMA secure scheme in the standard model. Before presenting our construction, we need to recap the underlying bonsai tree concept from [CHKP10].

7.2.1 Bonsai Trees

The notion of "bonsai trees" on lattices is introduced in [CHKP10] in analogy to arboriculture. An arborist always starts with a certain amount of *undirected*, i.e., random, natural growth that he cannot control. Then, he applies his tools and starts cultivating individual branches to achieve the desired looks via *directed* growth. The arborist is successful if the resulting tree still looks sufficiently natural to the observer. Once cultivated, a branch can easily be *extended* to form more directed growth without too much additional care. Instead of extending directed growth, the

arborist can also generate a *randomized* offsprings, which can be given to another arborist who can easily cultivate them by *extending* growth. The offsprings hide the first arborist's work and the employed techniques. We formalize these concepts in the context of lattices. A (binary) bonsai tree is generated out of a root $\mathbf{A}$ and branches $\mathbf{B}_i^{(b)} \in \mathbb{Z}_q^{n \times m_i}$, $b \in \{0,1\}$, $m_i \leq \text{poly}(n)$, $i \leq k \leq \text{poly}(n)$, that are statistically close to uniform. The entire tree is the set $\{\mathbf{A} \| \mathbf{B}_1^{(x_1)} \| \cdots \| \mathbf{B}_k^{(x_k)} : \mathbf{x} \in \{0,1\}^k\}$.

Proposition 7.2 (Directed Growth)**.** *Let $C > 0$ and $\delta > 0$ be constants and let $q \geq 3$ be odd. There is a PPT algorithm $\mathsf{ExtLattice}(\mathbf{A}_1, m_2)$ that, given uniformly random $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times m_1}$ for any $m_1 \geq (1+\delta) n \log_2(q)$ and $\text{poly}(n)$-bounded $m_2 \geq (4+2\delta) n \log_2(q)$, outputs $(\mathbf{A}_2 \in \mathbb{Z}_q^{n \times m_2}, \mathbf{S} \in \mathbb{Z}^{m \times m})$, where $m = m_1 + m_2$, such that $\mathbf{A} = \mathbf{A}_1 \| \mathbf{A}_2$ is within statistical distance $m_2 q^{-\delta n/2}$ from uniform; $\mathbf{S}$ is a basis of $\Lambda_q^\perp(\mathbf{A}_1 \| \mathbf{A}_2)$; $\|\mathbf{S}\| \leq L = Cn \log_2(q)$ with overwhelming probability; and $\left\|\tilde{\mathbf{S}}\right\| \leq \tilde{L} = 1 + C\sqrt{(1+\delta) n \log_2(n)} \leq 1 + C\sqrt{m_1}$ with overwhelming probability.*

The proposition reflects the most recent result on trapdoors for q-ary lattices (cf. Proposition 2.4). An interpretation in terms of arboriculture is generating "directed growth" out of "undirected growth" because one starts with some random growth $\mathbf{A}_1$ and cultivates a branch $\mathbf{A}_1 \| \mathbf{A}_2$ along with a trapdoor $\mathbf{S}$, which is the arborist's journal or a trace of his work. However, the observer cannot distinguish undirected growth from directed growth.

A central observation is that knowing a trapdoor for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ implies knowing a trapdoor for all $\mathbf{A}' \in \mathbb{Z}_q^{n \times m'}$, $m' \geq m$, when the columns of $\mathbf{A}$ form a prefix for the sequence of columns in $\mathbf{A}'$.[1] We write $\mathbf{A} \sqsubset \mathbf{A}'$. This is because one can apply the trapdoor in dimension m and then pad the resulting vector with zeros to solve SIS in dimension m'. Another option is to derive an actual trapdoor for the such a superlattice as follow.

Proposition 7.3 (Extending Control)**.** *There is deterministic polynomial time algorithm $\mathsf{ExtBasis}(\mathbf{S}_1, \mathbf{A} = \mathbf{A}_1 \| \mathbf{A}_2)$ that takes a basis $\mathbf{S}$ of $\Lambda_q^\perp(\mathbf{A}_1)$ and a matrix $\mathbf{A}$ with $\mathbb{Z}_q^{n \times m_1} \ni \mathbf{A}_1 \sqsubset \mathbf{A} \in \mathbb{Z}_q^{n \times (m_1+m_2)}$ as input. If $m_1 \geq 2n \log_2(q)$, it outputs a basis $\mathbf{S}$ for $\Lambda_q^\perp(\mathbf{A})$ with $\left\|\tilde{\mathbf{S}}\right\| = \left\|\tilde{\mathbf{S}}_1\right\|$.*

Although the following concept of trapdoor delegation is not used here, we include it for the sake of completeness. It is important when extending our construction to the hierarchical identity-based setting as described in [Rüc10c]. Whenever trapdoor

[1] Actually, any subset of columns works when a suitable permutation is applied.

delegation is required, one cannot simply use "extending control" and hand over the resulting basis as it leaks information about the original trapdoor. We need to re-randomize the source trapdoor first.

Proposition 7.4 (Randomizing Control)**.** *On input a basis* $\mathbf{S}$ *of the lattice* $\Lambda_q^\perp(\mathbf{A})$ *of dimension* m *and a Gaussian parameter* $\eta \geq \left\|\tilde{\mathbf{S}}\right\| \theta(n)$ *for* $\theta(n) = \omega(\sqrt{\log(n)})$*, the polynomial time algorithm* $\mathsf{RandBasis}(\mathbf{S}, \eta)$ *outputs a basis* $\mathbf{S}'$ *of* $\Lambda_q^\perp(\mathbf{A})$ *with* $\left\|\tilde{\mathbf{S}}'\right\| \leq \eta\sqrt{m}$*. The basis is independent of* $\mathbf{S}$ *in the sense that for any two bases* $\mathbf{S}_0, \mathbf{S}_1$ *of* $\Lambda_q^\perp(\mathbf{A})$ *and* $\eta \geq \max\{\left\|\tilde{\mathbf{S}}_0\right\|, \left\|\tilde{\mathbf{S}}_1\right\|\}\theta(n)$*,* $\mathsf{RandBasis}(\mathbf{S}_0, \eta)$ *is within negligible statistical distance of* $\mathsf{RandBasis}(\mathbf{S}_1, \eta)$*.*

7.2.2 Our Construction

In the following, we let $\mathsf{H} \in \mathcal{H}(1^\lambda)$ be a collision-resistant hash function that maps into the message space $\{0,1\}^\lambda$. For simplicity, let us assume that finding collisions in polynomial time is possible with probability at most $\epsilon_{\mathcal{H}}$. Given the tools from the previous section and PSTF from Section 2.7.3, our scheme $\mathrm{DS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ works as follows.

Parameters: Let $q, \tilde{L}, m_1, m_2$ be chosen according to Proposition 7.2 and let $\eta = \tilde{L}\theta(n)$ for any $\theta(n) = \omega(\sqrt{\log(n)})$ and $d \leftarrow \eta\sqrt{m_1 + (\lambda+1)m_2}$. These parameters may be excluded from the public key as they are the same for all users.

Key Generation: $\mathsf{Kg}(1^n)$ samples $\mathbf{A}_1 \leftarrow_\$ \mathbb{Z}_q^{n \times m_1}$ and uses $\mathsf{ExtLattice}(\mathbf{A}_1, m_2)$ to generate a description $\mathbf{A} \in \mathbb{Z}_q^{n \times (m_1+m_2)}$ of the master lattice $\Lambda_q^\perp(\mathbf{A})$ together with a trapdoor $\mathbf{S}$ such that $\left\|\tilde{\mathbf{S}}\right\| \leq \tilde{L}$. Furthermore, it picks a set $\langle \mathbf{B} \rangle := \left((\mathbf{B}_i^{(0)}, \mathbf{B}_i^{(1)})\right)_{i=1}^{\lambda}$ of random matrices in $\mathbb{Z}_q^{n \times m_2}$ as well as $\mathbf{y} \leftarrow_\$ \mathbb{Z}_q^n$. The output is a secret key $\mathbf{S}$ and a public key $(\mathbf{A}, \langle \mathbf{B} \rangle, \mathbf{y})$.

Signing: $\mathsf{Sign}(\mathbf{S}, msg \in \{0,1\}^*)$ selects $r \leftarrow_\$ \{0,1\}^n$, computes $h \leftarrow \mathsf{H}(msg, r)$ as well as the signature $\sigma \leftarrow \mathsf{SamplePre}(\mathbf{S}_h, \eta, \mathbf{y})$. The trapdoor $\mathbf{S}_h$ is formed via $\mathsf{ExtBasis}(\mathbf{S}, \mathbf{A}_h)$, where $\mathbf{A}_h := \mathbf{A} \| \mathbf{B}_1^{(h_1)} \| \cdots \| \mathbf{B}_\lambda^{(h_\lambda)}$. The output is (σ, r).

Verification: $\mathsf{Vf}((\mathbf{A}, \langle \mathbf{B} \rangle, \mathbf{y}), (\sigma, r), msg)$ outputs 1 if and only if $\|\sigma\|_2 \leq d$ and $\mathbf{A}_h \sigma \equiv \mathbf{y}$ for $h \leftarrow \mathsf{H}(msg, r)$.

The scheme is complete because all signatures are generated using a basis of length $\tilde{L}$ and with the Gaussian parameter $\eta = \omega(\sqrt{\log(n)})\tilde{L}$. The total dimension is $m = m_1 + (\lambda + 1)m_2$. Thus, $\mathsf{SamplePre}$ outputs signatures of length at most $\eta\sqrt{m_1 + (\lambda+1)m_2} = d$ that are accepted by Vf. In order to get the full (SU-CMA) scheme, we wrap the hash value h with a chameleon hash function.

For improved storage efficiency, the construction can be instantiated with ideal lattices as well. Refer to the next chapter for a discussion of the required tools to do so.

Security We prove that DS is SU-SMA secure. Let $T_{\mathsf{func}} = T_{\mathsf{func}}(n)$ be the cost function for the function func and let $T_{\mathsf{List}} = T_{\mathsf{List}}(n)$ be the cost function for list processing, which is explained in the analysis below.

Theorem 7.5. *Let $\theta(n) = \omega(\sqrt{\log(n)})$. DS is $(t, Q_{\mathsf{OSign}}, \epsilon)$ strongly unforgeable under static message attacks (SU-SMA) if SIS(n, m, q, ν) with norm bound $\nu = 2\tilde{L}\theta(n)\sqrt{m_1 + (\lambda+1)m_2}$ is $(t + \lambda Q_{\mathsf{OSign}} T_{\mathsf{List}} + T_{\mathsf{ExtLattice}} + Q_{\mathsf{OSign}}(T_{\mathsf{SamplePre}} + T_{\mathsf{ExtBasis}}), 1/2(1 - n^{-\omega(1)})(\epsilon - Q_{\mathsf{OSign}}^2/2^n - \epsilon_{\mathcal{H}})/(\lambda \cdot Q_{\mathsf{OSign}}))$-hard.*

The idea is to separate the adversaries into two classes. One works in the EU-SMA sense and the other exploits the additional freedom of the SU-SMA setting. The reduction guesses the type of adversary before handing over the public key. If it expects an EU-SMA forger, the reduction knows $\mathbf{x}$ with $\mathbf{A}\mathbf{x} \equiv \mathbf{y}$ and forces the forger to solve an inhomogeneous SIS, for which the reduction does not know the trapdoor. Together with $\mathbf{x}$, it can solve the corresponding SIS with overwhelming probability. For the SU-SMA forger, the reduction has to guess the index i^* of the signature query that will be recycled in the forgery. This can be done with probability at least $1/Q_{\mathsf{OSign}}$. There, it plants an $\mathbf{x}$ with $\mathbf{A}_{\mathsf{H}(msg_{i^*}, r_{i^*})}\mathbf{x} \equiv \mathbf{y}$. Again, with the adversary's help, the reduction solves SIS with overwhelming probability, while being able to answer a single signature query for msg_{i^*} with $\mathbf{x}$.

Proof. We assume that there is a successful adversary $\mathcal{A}$ against SU-SMA unforgeability of DS and we construct a reduction $\mathcal{B}^{\mathcal{A}}$ that solves SIS. The reduction receives a list of messages and returns a list of signatures along with the public key. Then, the adversary either outputs a weak forgery (EU-SMA) or a strong forgery (SU-SMA). The reduction guesses the type of adversary beforehand and solves SIS is both cases.

Setup: On input $\mathbf{A}^\star = \mathbf{A}\|\mathbf{U}_1^{(0)}\|\mathbf{U}_1^{(1)}\|\cdots\|\mathbf{U}_\lambda^{(0)}\|\mathbf{U}_\lambda^{(1)} \in \mathbb{Z}_q^{m_1+(2\lambda+1)m_2}$ from the SIS problem, the reduction invokes $\mathcal{A}$ and receives a list $(msg_i)_{i=1}^{Q_{\mathsf{OSign}}}$ of messages. It flips a coin $c \leftarrow_{\$} \{0,1\}$ (0=EU, 1=SU) and picks $i^* \leftarrow_{\$} [Q_{\mathsf{OSign}}]$.

Then, it chooses $r_i \leftarrow_{\$} \{0,1\}^n$ and computes $h^{(i)} \leftarrow \mathsf{H}(msg_i, r_i)$ for all $i \in [Q_{\mathsf{OSign}}]$. If there is a collision under H, the reduction starts over. Let $\langle \pi \rangle := \left(\pi^{(i)}\right)_{i=1}^{p}$ be the set of all strings $\pi \in \{0,1\}^{\leq \lambda}$ such that $\pi \not\sqsubset h^{(j)}$ for $j \in [Q_{\mathsf{OSign}}] \setminus \{c \cdot i^*\}$, and $\pi^{(i)} \not\sqsubset \pi^{(j)}$ for all distinct pairs $(\pi^{(i)}, \pi^{(j)})$ in $\langle \pi \rangle$. The set $\langle \pi \rangle$ contains $p \leq \lambda \cdot Q_{\mathsf{OSign}}$ elements. Now, randomly select an element $\pi \leftarrow_{\$} \langle \pi \rangle$, which will represent the challenge subtree, i.e., the subtree of the bonsai tree, where we inject the input from the SIS problem. Let $l_\pi \leftarrow |\pi|$ be the bit length of π.

To embed the challenge, $\mathcal{B}$ sets up matrices $\mathbf{B}_i^{(\pi_i)} \leftarrow \mathbf{U}_i^{(0)}$ for $i \in [l_\pi]$ and matrices $\mathbf{B}_i^{(b)} \leftarrow \mathbf{U}_i^{(b)}$ for $b \in \{0,1\}$ and $i = l_\pi + 1, \ldots, \lambda$. Then, $\mathcal{B}$ "rigs" the public key so that it is able to answer signature queries for all subtrees but π: compute $\mathbf{B}_i^{1-\pi_i}$ and $\mathbf{S}_i$ via $\mathsf{ExtLattice}(\mathbf{A} \| \mathbf{B}_1^{(\pi_1)} \| \cdots \| \mathbf{B}_{i-1}^{(\pi_{i-1})}, m_2)$ for $i \in [l_\pi]$.

If $c = 0$, use $\mathsf{SampleDom}$ with parameter η to sample a vector $\mathbf{x} \in \mathbb{Z}^{m_1+m_2}$ and compute $\mathbf{y} \leftarrow \mathbf{A}\mathbf{x} \bmod q$. For $c = 1$, sample $\mathbf{x} \in \mathbb{Z}^{m_1+(\lambda+1)m_2}$, and set $\mathbf{y} \leftarrow \mathbf{A}_{h^{(i^*)}}\mathbf{x} \bmod q$.

Signing: For all $i \neq c \cdot i^*$, let j be the smallest index with $h_j^{(i)} \neq \pi_j$. The reduction computes the signature $\sigma_i \leftarrow \mathsf{SamplePre}(\mathsf{ExtBasis}(\mathbf{S}_j, \mathbf{A}_{h^{(i)}}), \eta, \mathbf{y})$. For $i = c \cdot i^*$, the reduction sets $\sigma_i \leftarrow \mathbf{x}$.

The public key comprises $\mathbf{A}$, $\mathbf{y}$, and $\langle \mathbf{B} \rangle := \left((\mathbf{B}_i^{(0)}, \mathbf{B}_i^{(1)})\right)_{i=1}^{\lambda}$ and the reduction returns the public key and the list $((\sigma_i, r_i))_{i=1}^{Q_{\mathsf{OSign}}}$ of signatures to $\mathcal{A}$.

Output: Eventually, $\mathcal{A}$ outputs its forgery (msg^*, σ^*, r^*). If $c = 0$, the reduction pads $\mathbf{x}$ with zeros to form $\mathbf{x}' \in \mathbb{Z}^{m_1+(\lambda+1)m_2}$ and outputs $\mathbf{x}' - \sigma^*$. In case $c = 1$, the reduction outputs $\mathbf{x} - \sigma^*$. In either case, the output needs to be suitably padded and rearranged to solve SIS for $\mathbf{A}^\star$.

Analysis First of all, the setup phase is efficient. As in [CHKP10], the set $\langle \pi \rangle$ is of polynomial size in n. Finding each of the strings π in the set costs T_{List}. If $\mathcal{A}$ runs in time t, then the reduction runs in time $t + \lambda \cdot Q_{\mathsf{OSign}} T_{\mathsf{List}} + T_{\mathsf{ExtLattice}} + Q_{\mathsf{OSign}}(T_{\mathsf{SamplePre}} + T_{\mathsf{ExtBasis}})$ plus some minor overhead. Next, notice that the setup phase outputs valid signatures and a public key that is indistinguishable from uniform. The only deviation from the real scenario is that there might be a collision under H, which happens with probability $\leq \epsilon_{\mathcal{H}}$ in $\mathsf{Exp}^{\mathsf{CR}}_{\mathcal{A},\mathcal{H}}$ (cf. 2.3.1). It is also possible that the signer chooses the same r when queried with a message msg twice. This happens with probability at most $q_{\mathsf{OSign}}^2/2^n$.

As for the output (msg^*, σ^*, r^*) of $\mathcal{A}$, one of two things happens with probability 1/2: There is an index i with $h^* = \mathsf{H}(msg^*, r^*) = h^{(i)}$. Then, the reduction must have guessed $i^* = i$ correctly (probability $\geq 1/Q_{\mathsf{OSign}}$) and $\mathcal{A}$ outputs a signature $\sigma^* \neq \mathbf{x}$. If there is no such i, the reduction works if $\pi \sqsubset h^*$ (probability $1/(\lambda \cdot Q_{\mathsf{OSign}})$) and the padded $\mathbf{x}' \neq \sigma^*$ (probability $1 - n^{-\omega(1)}$). Therefore, the total success probability ϵ' is at least $1/2(\epsilon - q_{\mathsf{OSign}}^2/2^n - \epsilon_{\mathcal{H}})(1 - n^{-\omega(1)})/(\lambda \cdot Q_{\mathsf{OSign}})$. □

Via Proposition 2.1, we arrive at a reduction from worst-case lattices problems and obtain similar results for SU-CMA via Theorem 7.1.

Corollary 7.6. *DS is strongly unforgeable under static message attacks (SU-SMA) if if SIVP is hard in the worst case for approximation factors* $\gamma = \widetilde{\mathcal{O}}(\sqrt{\lambda} n \sqrt{n})$.

7.3 Conclusion and Open Problems

We have constructed the first treeless, standard model signature scheme from lattices that is secure in the strong unforgeability model. With our modification to [CHKP10], we have shown once more that working over cosets instead of over lattices themselves is very useful in security proofs.

Unfortunately, all known lattice-based signature schemes in the standard model are rather impractical because their use of trapdoors and sampling. Constructing such a scheme without trapdoors remains a major open research problem. Even if, one day, sampling becomes more efficient and trapdoors become smaller, a trapdoorless scheme would still have its benefits because all users could share the same lattice.

Chapter 8

Identity-based Identification

Identification schemes are one of the most important primitives in modern cryptography because typical e-business or e-government applications essentially rely on secure online access control. With identity-based identification schemes (IBI), motivated by Shamir [Sha84], one can get rid of public-key infrastructures, which are unfavorable in today's widespread decentralized networks. The public key is replaced with a unique identifier string, such as an e-mail address, and the secret key is "extracted" by a trusted party for this identifier.

In the strongest security model ADAPT-ID-IMP-CA [KH05, BNN09], the adversary has access to provers for arbitrary identities in a training phase. The provers may be accessed concurrently. In addition, the adversary may query a secret-key extraction oracle to corrupt identities of its choice. Then, it outputs a non-corrupted challenge identity and wins if it is able to impersonate it. Weaker attacks include impersonation under active (non-concurrent) attacks (-AA), or passive attacks (-PA) without any direct prover access.

It is well-known that identity-based identification schemes can be realized in the standard model with a so-called certification approach due to Bellare, Neven, and Namprempre [BNN09] but these generic, black-box constructions require a certain computational and bandwidth overhead. The only known direct construction from lattices is sketched in [SSTX09] and requires random oracles.

Our Contribution We propose a new, modular approach for designing identity-based identification schemes. It is yet another application of the "hash-sign-switch" trick using chameleon hash functions [KR00]. Therefore, we introduce a weaker security model STAT-ID-IMP-CA related to the static message attack model (cf. Section 7.1) for signature schemes. Thus, the ADAPT-ID-IMP-CA model is weakened by forcing the adversary to output the list of identities to corrupt before seeing the master public key of the key extraction authority. Such constructions are potentially easier to achieve and simpler to design.

Using chameleon hash functions, we prove the following theorem for identification schemes, but it also holds for other identity-based constructions. Namely, whenever a message is sent from the secret-key holder to the receiver, which is why it gives rise to a strongly unforgeable identity-based signature scheme in the standard model using Chapter 7 and additional techniques from [Rüc10c]. The sender simply appends the randomness for the chameleon hash to his or her transmission. Then, the receiver can compute the quasi-public-key using identity and randomness.

Theorem (Generic Conversion)**.** *STAT-ID-IMP-CA (-PA / -AA) implies ADAPT-ID-IMP-CA (-PA / -AA) if chameleon hash functions exist.*

Using lattices, we propose an exemplary instantiation IBI of our STAT-ID-IMP-CA model. We present it using ideal lattices because its presentation bears a considerable similarity with our blind signature scheme in Chapter 4 as both are based on Lyubashevsky's witness-indistinguishable proof of knowledge [Lyu08a, Lyu08b].

Theorem (Main Construction)**.** *Assuming the existence of chameleon hash functions with range* $\{0,1\}^\lambda$*, let* $D \subseteq \mathbf{R}_0$ *such that* $\mathbf{f} \in D$ *if* $\|\mathbf{f}\|_\infty = \widetilde{\mathcal{O}}(\lambda n^2\sqrt{n})$*. IBI is secure in the STAT-ID-IMP-CA model if the collision problem* $COL(\mathcal{H}(\mathbf{R}, \widetilde{\mathcal{O}}(\lambda)), D)$ *is hard.*

The Simulation Trick An essential part of our construction, and a contribution of independent interest, is a universal simulation technique for hierarchies of lattices, such as bonsai trees [CHKP10].

Consider an m_1-dimensional lattice $\Lambda_q^\perp(\mathbf{A})$, a vector $\mathbf{S} \leftarrow_\$ \mathbb{Z}_q^n$, and a list of m_2-dimensional super lattices $\Lambda_q^\perp(\mathbf{B}_1), \ldots, \Lambda_q^\perp(\mathbf{B}_k)$ with $\mathbf{A} \sqsubset \mathbf{B}_i$ for all $i \in [k]$. The task is to prove knowledge of a short vector $\mathbf{s}_i$ such that $\mathbf{B}_i\mathbf{s}_i \equiv \mathbf{S}$ for a particular i with a witness indistinguishable proof.

Furthermore, consider a simulation of the above provers. The simulator can sample $\mathbf{s}_0^\star \leftarrow \mathsf{SampleDom}(m_1, \eta)$ and set $\mathbf{S} \leftarrow \mathbf{A}\mathbf{s}_0^\star \bmod q$. Then, the proofs for all i can be simulated efficiently without applying SampleDom or SamplePre again. For identity-based identification, this means that the simulator does not need to run the key extraction algorithm and it does not have to know a trapdoor. Instead, it can use the 0-padded universal pseudo-secret $\mathbf{s}^\star \leftarrow \mathbf{s}_0^\star \| \mathbf{0} \in \mathbb{Z}^{m_2}$ in all proofs and witness-indistinguishability hides this deviation.

Organization After introducing the new, weaker model for identity-based identification in Section 8.1 that is akin to static message attacks for signatures, we discuss its relation to the standard security notion. In Section 8.1.2, we essentially apply the "hash-sign-switch" paradigm of [KR00] to prove that our weak security notion is equivalent to the standard notion. Then, we use an ideal-lattice interpretation of bonsai trees [CHKP10] to instantiate our model in Section 8.2 before we conclude the chapter in Section 8.3.

In this chapter, we reprint the main construction in [Rüc10a], of which the dissertation author was the main investigator and author.

8.1 From Static to Adaptive Identity Security

One of the contributions in this work is a method to simplify the construction of identity-based identification schemes in the strongest security model. We discuss this model in the following and propose a slightly weaker version. Then, we prove that they are equivalent if chameleon hash functions exist. The transformation is tightly related to the conversion described in Section 7.1.

8.1.1 Security Model

Identity-based identification schemes IBI = $(\mathsf{Kg}, \mathsf{Extract}, \mathsf{Protocol})$ are specified as follows.

Master-key Generation: $\mathsf{Kg}(1^n)$ outputs a master secret key msk and a master public key mpk.

Key Extraction: $\mathsf{Extract}(msk, \mathsf{ID})$ extracts a secret key sk_{ID} for the given identifier ID.

Identification Protocol: $\mathsf{Protocol}$ is an interactive protocol between a prover algorithm $\mathcal{P}_{\mathsf{ID}}(mpk, sk_{\mathsf{ID}})$ and a verifier $\mathcal{V}(mpk, \mathsf{ID})$, where $\mathcal{P}_{\mathsf{ID}}$ convinces $\mathcal{V}$ of its identity. The joint execution $\langle \mathcal{P}_{\mathsf{ID}}, \mathcal{V} \rangle$ yields a bit b. If $b = 1$, $\mathcal{V}$ accepts and if $b = 0$, the verifier rejects.

The security model for identity-based identification [Sha84] was formalized by Kurosawa and Heng [KH05] and it is also discussed in the recent work of Bellare, Neven, and Namprempre [BNN09]. Security is proven against impersonation under adaptive identity attacks as described in the ADAPT-ID-IMP-CA experiment. The adversary (impersonator) $\mathcal{A}$ works in two modes: *m_verify* and *m_impersonate*. In mode *m_verify*, it has access to mpk, to a secret key extraction oracle $\mathsf{OExtract}$, and to provers $\mathcal{P}_{\mathsf{ID}}$ for arbitrary identities ID. The adversary may query the provers concurrently, i.e., run multiple, arbitrarily interleaved sessions with provers that have individual random tapes. At some point, it selects a target identity ID^*, which it tries to impersonate in the second phase. In mode *m_impersonate*, $\mathcal{A}$ has access to provers and secret keys for all identities $\neq \mathsf{ID}^*$ and it is supposed to convince an honest verifier that it knows the secret key for ID^*. Obviously, the secret key for ID^* must not have been among the queries to the extraction oracle in the first phase. Also, note that $\mathcal{A}$ is allowed to keep a state *s_verify* across modes.

Experiment $\mathsf{Exp}^{\mathsf{ADAPT\text{-}ID\text{-}IMP\text{-}CA}}_{\mathcal{A},\mathrm{IBI}}(n)$
$(msk, mpk) \leftarrow \mathrm{IBI}.\mathsf{Kg}(1^n)$
$(\mathsf{ID}^*, s\_verify) \leftarrow \mathcal{A}^{\langle \mathcal{P}_{\mathsf{ID},\cdot}\rangle^\infty, \mathsf{OExtract}(msk,\cdot)}(m\_verify, mpk)$
Let $(\mathsf{ID}_i)_{i=1}^Q$ be the ID's queried to $\mathsf{OExtract}$.
$b \leftarrow \left\langle \mathcal{A}^{\langle \mathcal{P}_{\neq \mathsf{ID}^*,\cdot}\rangle^\infty, \mathsf{Extract}_{\neq \mathsf{ID}^*}(msk,\cdot)}, \mathcal{V} \right\rangle ((m\_impersonate, s\_verify), \mathsf{ID}^*)$
Return 1 iff $b = 1 \wedge \mathsf{ID}^* \notin (\mathsf{ID}_i)_{i=1}^Q$

In the following, we propose a relaxed security model, called security against concurrent identity-based impersonation under *static* identity attacks (STAT-ID-IMP-CA). The model gives $\mathcal{A}$ significantly less power as the adversary, in mode *m_find*, has to submit a list of distinct identities to the oracle $\mathsf{OExtract}$ before seeing the master public key. It then receives the extracted secret keys together with *mpk* and does not have access to $\mathsf{OExtract}$ anymore. Again, the adversary is allowed to keep a state across modes.

Experiment $\mathsf{Exp}^{\mathsf{STAT\text{-}ID\text{-}IMP\text{-}CA}}_{\mathcal{A},\mathrm{IBI}}(n)$
$(\mathsf{ID}_1, \ldots, \mathsf{ID}_Q, s\_find) \leftarrow \mathcal{A}(m\_find)$ for distinct ID_i
$(msk, mpk) \leftarrow \mathrm{IBI}.\mathsf{Kg}(1^n)$
$sk_i \leftarrow \mathsf{OExtract}(msk, \mathsf{ID}_i)$ for $i \in [Q]$
$(\mathsf{ID}^*, s\_verify) \leftarrow \mathcal{A}^{\langle \mathcal{P}_{\mathsf{ID},\cdot}\rangle^\infty}(m\_verify, s\_find, mpk, (sk_i)_{i=1}^Q)$
$b \leftarrow \left\langle \mathcal{A}^{\langle \mathcal{P}_{\neq \mathsf{ID}^*,\cdot}\rangle^\infty}, \mathcal{V} \right\rangle ((m\_impersonate, s\_verify), \mathsf{ID}^*)$
Return 1 iff $b = 1 \wedge \mathsf{ID}^* \notin (\mathsf{ID}_i)_{i=1}^Q$

We have revisited a similar weak notion for signatures in Chapter 7. There, we have also seen that chameleon hash functions can be used to amplify the security of a given, weakly secure scheme. In the following, we show that essentially the same transformation can be used to strengthen the security of identification schemes. More precisely, we provide a black-box transformation from STAT-ID-IMP-CA to ADAPT-ID-IMP-CA security in the standard model. This makes the construction of such scheme more modular and potentially more efficient. In any case, the proofs are greatly simplified because one can prepare for all key extraction queries before handing over the master public key.

Note that the weaker security notions for identification schemes, passive (PA) and active (AA), apply as well and the experiments in this section can be easily changed to cover these attacks. All definitions carry over to the hierarchical setting [GS02], where identities can be concatenated to describe a subordinate identity and

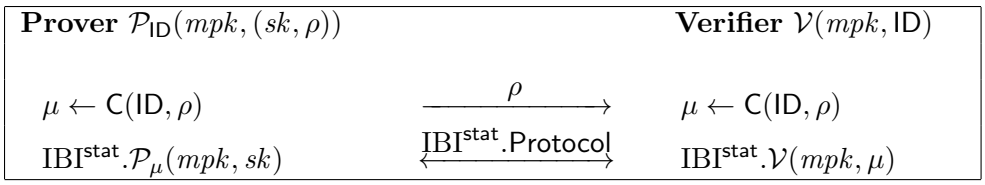


Figure 8.1: Identity-based identification protocol for $\mathrm{IBI}^{\mathsf{adapt}}$.

its relation in an organizational structure. Here, every entity can act as a key extraction authority for its subordinates.

8.1.2 From STAT-ID-IMP-CA to ADAPT-ID-IMP-CA

We propose a generic, black-box transformation from static-identity security to adaptive-identity security. Notice that the transformation is property-preserving with regard to the identification scheme, i.e., security under passive, active, and concurrent attacks carries over.

Theorem 8.1 (Transformation)**.** *STAT-ID-IMP-CA (-PA / -AA) implies ADAPT-ID-IMP-CA (-PA / -AA) if chameleon hash functions exist.*

Proof. Suppose we have a scheme $\mathrm{IBI}^{\mathsf{stat}} = (\mathsf{Kg},\mathsf{Extract},\mathsf{Protocol})$ that is secure against static identity attacks, we show how to construct a scheme $\mathrm{IBI}^{\mathsf{adapt}} = (\mathsf{Kg}',\mathsf{Extract}',\mathsf{Protocol}')$ that is secure against adaptive identity attacks using a family $\mathcal{C} = (\mathsf{Kg},\Psi)$ of chameleon hash functions.

Master-key Generation: $\mathsf{Kg}'(1^n)$ runs $(msk,mpk) \leftarrow \mathrm{IBI}^{\mathsf{stat}}.\mathsf{Kg}(1^n)$ and select a chameleon hash function $(\mathsf{C},\cdot) \leftarrow \mathcal{C}.\mathsf{Kg}(1^n)$. It returns the secret key msk and the public key $mpk' \leftarrow (mpk,\mathsf{C})$.

Key Extraction: $\mathsf{Extract}'(msk,\mathsf{ID})$ picks $\rho \leftarrow_{\Psi} \mathcal{R}$ and computes $\mu \leftarrow \mathsf{C}(\mathsf{ID},\rho)$. Then, it computes the secret key for the pseudo identity μ via $sk \leftarrow \mathrm{IBI}.\mathsf{Extract}(msk,\mu)$. The algorithm returns the pair (sk,ρ).

Identification Protocol: $\mathsf{Protocol}'$ uses $\mathrm{IBI}^{\mathsf{stat}}.\mathsf{Protocol}$ as a sub-protocol after re-computing the randomized pseudo identity $\mu \leftarrow \mathsf{C}(\mathsf{ID},\rho)$. See Figure 8.1.

First of all notice that the chameleon hash function prevents the adversary from reusing a given secret key sk for a particular identity ID to impersonate a different identity $\mathsf{ID}^* \neq \mathsf{ID}$. Such an adversary would refute the collision resistance of the

family $\mathcal{C}$. The reduction is straightforward. Therefore, we focus on the reduction $\mathcal{B}^{\mathcal{A}}$ in the presence of an impersonator $\mathcal{A}$ that does not exploit any weakness in the chameleon hash function. Suppose that the adversary makes at most Q queries to the extraction oracle. The reduction plays in the STAT-ID-IMP-CA experiment and simulates $\mathcal{A}$'s environment as in the ADAPT-ID-IMP-CA environment. $\mathcal{B}$ has access to an extraction oracle before seeing the master public key from $\mathrm{IBI}^{\mathsf{stat}}$ and to arbitrary provers for $\mathrm{IBI}^{\mathsf{stat}}$.

Setup: $\mathcal{B}$ chooses a chameleon hash function $(\mathsf{C}, \mathsf{C}^{-1}) \leftarrow \mathcal{C}.\mathsf{Kg}(1^n)$. It picks a set of randomized identities $\mu_1, \ldots, \mu_Q \leftarrow_{\$} \mathcal{I}$. Afterwards, $\mathcal{B}$ calls its extraction oracle for $\mathrm{IBI}^{\mathsf{stat}}.\mathsf{Extract}$ to obtain the corresponding secret keys $sk_1, \ldots, sk_Q$ and sets up a counter $\imath \leftarrow 0$. It runs $\mathcal{A}$ on input (mpk, C).

Extraction Queries: Whenever $\mathcal{A}$ queries an identity ID to its extraction oracle, the internal counter $\imath$ is incremented and the reduction calls $\rho \leftarrow \mathsf{C}^{-1}(\mu_\imath, \mathsf{ID})$. It returns $(sk_\imath, \rho)$.

Prover Queries: The simulator runs the protocol in Figure 8.1, using its access to external provers for $\mathrm{IBI}^{\mathsf{stat}}$.

Impersonation Attempt: At some point, $\mathcal{A}$ outputs a challenge identity ID^*, which has not been queried to the extraction oracle before. From this point on, the extraction oracle answers $\perp$ when queried with ID^*. When the adversary instantiates a verifier to prove its identity ID^* with randomness ρ^*, $\mathcal{B}$ forwards all messages to and from its external challenge verifier in the STAT-ID-IMP-CA experiment for target identity $\mu^* = \mathsf{C}(\mathsf{ID}^*, \rho^*)$.

The environment of $\mathcal{A}$ is perfectly simulated if the input-output relation of $\mathcal{C}$ can be sampled perfectly. The extraction oracle in the simulation has not been queried with ID^*, so $\mathcal{B}$ has never called its extraction oracle with identity $\mu^* = \mathsf{C}(\mathsf{ID}^*, \rho^*)$ (but with negligible probability). If $\mathcal{A}$ is successful in the impersonation attempt, so is $\mathcal{B}$. □

In principle, our transform works for all *authentication-type* ID-based cryptography, e.g., ID-based identification or signatures. In particular, it can be used to construction a strongly-unforgeable identity-based signature scheme in the standard model from our construction in Chapter 7.

For identity-based encryption, this does not work because there is no message-flow from the secret-key holder to the public-key holder. In other words, the encrypting

party cannot derive the recipients full public key as it does not know the randomness for the chameleon hash.

8.2 An Instantiation with Lattices

Using the transformation from the previous section, we construct an adaptively secure identity-based identification scheme in the standard model. Our construction reuses large parts of the machinery from Chapter 4 as well as from Chapter 7. It uses the following ideal-lattice interpretation of bonsai trees.

8.2.1 Convoluted Bonsai Trees

In Section 7.2.1, we have seen the bonsai tree principle from [CHKP10] in the setting of q-ary lattices. There, the authors also point out that bonsai trees from ideal lattices seem possible. We confirm this observation by making it explicit, based on the following family PSTF = (Kg, Eval, SampleDom, SamplePre) of preimage sampleable trapdoor functions in ideal lattices.

Parameters: The following parameters are functions in n. $q = \mathrm{poly}(n)$, $m = \Omega(\log(n))$, $\tilde{L} = \widetilde{\mathcal{O}}(\sqrt{n})$, and $\eta = \theta(n)\tilde{L}$ for $\theta(n) = \omega(\sqrt{\log(n)})$.

Key Generation [SSTX09]: $\mathsf{Kg}(1^n)$ outputs a public key $\hat{\mathbf{a}} \in \mathbf{R}^m$ and a secret trapdoor matrix $\mathbf{T} \in \mathbb{Z}^{mn \times mn}$ that is a basis of $\Lambda^{\perp}_{\mathbf{R}}(\hat{\mathbf{a}})$. Furthermore, we have $\left\|\tilde{\mathbf{T}}\right\| \leq \tilde{L}$.

Evaluation: $\mathsf{Eval}(\hat{\mathbf{a}}, \hat{\mathbf{x}})$, on input $\hat{\mathbf{x}} \in \mathbf{R}_0^m$ returns $\hat{\mathbf{a}} \circledast \hat{\mathbf{x}} \bmod q$. Recall that $\hat{\mathbf{a}} \circledast \hat{\mathbf{x}} := \sum_{i=1}^{m} \mathbf{a}_i \mathbf{x}_i$.

Domain Sampling: $\mathsf{SampleDom}(m, \eta)$ returns $\hat{\mathbf{x}} \in \mathbf{R}_0^m$ with $0 < \|\hat{\mathbf{x}}\|_\infty \leq \eta\theta(m)$ with overwhelming probability.

Preimage Sampling: $\mathsf{SamplePre}(\mathbf{T}, \eta, \mathbf{Y})$ outputs $\hat{\mathbf{x}} \in \mathbf{R}_0^m$ subject to $\hat{\mathbf{a}} \circledast \hat{\mathbf{x}} \equiv \mathbf{Y}$ and $\|\mathbf{x}\|_\infty \leq \eta\theta(m)$ with overwhelming probability.

Essentially, the above family is the same as the one described in Section 2.7.3. The key generation differs and yields a public key $\hat{\mathbf{a}}$ that is smaller by a factor n.

As described earlier in Section 7.2.1, we need to implement ExtLattice (directed growth), ExtBasis (extending control), and RandBasis (randomizing control) for a bonsai tree. Its root is $\hat{\mathbf{a}}$ and its branches are generated by uniformly random $\hat{\mathbf{b}}_i^{(b)} \in$

$\mathbf{R}^{m_i}$, $b \in \{0,1\}, i \leq k \leq \mathrm{poly}(n)$. The entire tree is the set $\{\hat{\mathbf{a}} \| \hat{\mathbf{b}}_1^{(x_1)} \| \cdots \| \hat{\mathbf{b}}_k^{(x_k)} : \mathbf{x} \in \{0,1\}^k\}$. The core of the bonsai tree technique is the observation that we can append two vectors of polynomials $\hat{\mathbf{a}} \in \mathbf{R}^{m_1}$ and $\hat{\mathbf{b}} \in \mathbf{R}^{m_2}$ to form $\hat{\mathbf{c}} = \hat{\mathbf{a}} \| \hat{\mathbf{b}} \in \mathbf{R}^{m_1+m_2}$. Now, knowing a solution $\hat{\mathbf{x}} \in \mathbf{R}_0^{m_1}$ to the equation $\hat{\mathbf{a}} \circledast \hat{\mathbf{x}} \equiv \mathbf{0}$, we immediately obtain a solution $\hat{\mathbf{y}} \in \mathbf{R}_0^{m_1+m_2}$ to the equation $\hat{\mathbf{c}} \circledast \hat{\mathbf{y}} \equiv \mathbf{0}$ by setting $\hat{\mathbf{y}} = \hat{\mathbf{x}} \| \hat{\mathbf{0}} \in \mathbf{R}^{m_1+m_2}$ with $\|\hat{\mathbf{x}}\| = \|\hat{\mathbf{y}}\|$ for any norm. To see this, we directly apply the definition of $\circledast$ and obtain $\hat{\mathbf{c}} \circledast \hat{\mathbf{y}} = \hat{\mathbf{a}} \circledast \hat{\mathbf{x}} + \hat{\mathbf{b}} \circledast \hat{\mathbf{0}} = \mathbf{0}$.

Proposition 8.2 (Directed Growth)**.** *Let $r = \Omega(\log(n))$ and $3 \leq q = \mathrm{poly}(n)$ be a prime such that $q \equiv 3 \pmod 8$. There is a PPT algorithm $\mathsf{ExtLattice}(\hat{\mathbf{a}}, m)$ that, given a uniformly random $\hat{\mathbf{a}} \in \mathbf{R}^{m_1}$, with $m = m_1 + m_2 \geq (\lceil \log(q) \rceil + 1)(1+r)$ and $m_1 \geq 1$, generates $\hat{\mathbf{b}} \in \mathbf{R}^{m_2}$ with $m_2 = m - m_1$ together with a matrix $\mathbf{T} \in \mathbb{Z}^{mn \times mn}$.*

The algorithm succeeds with overwhelming probability and its outputs satisfies the following properties.

1. $\hat{\mathbf{a}} \| \hat{\mathbf{b}}$ *is within negligible statistical distance from uniform;*
2. $\|\mathbf{T}\| \leq L = \mathcal{O}(\sqrt{n \log(n)})$.

The proposition is an immediate consequence of [SSTX09, Theorem 3.1]. It does not guarantee a bound for $\left\|\tilde{\mathbf{T}}\right\|$ but we always have $\left\|\tilde{\mathbf{T}}\right\| \leq \|\mathbf{T}\|$.

Remember that we write $\hat{\mathbf{a}} \sqsubset \hat{\mathbf{c}}$ if there is a $\hat{\mathbf{b}}$ such that $\hat{\mathbf{c}} = \hat{\mathbf{a}} \| \hat{\mathbf{b}}$. Then, the following proposition allows us to extend a trapdoor for a lattice $\Lambda^{\perp}_{\mathbf{R}}(\hat{\mathbf{a}})$ to a trapdoor for any lattice $\Lambda^{\perp}_{\mathbf{R}}(\hat{\mathbf{c}})$ with $\hat{\mathbf{a}} \sqsubset \hat{\mathbf{c}}$.

Proposition 8.3 (Extending Control)**.** *There is a deterministic polynomial time algorithm $\mathsf{ExtBasis}(\mathbf{T}, \hat{\mathbf{c}} = \hat{\mathbf{a}} \| \hat{\mathbf{b}})$ that takes a basis $\mathbf{T}$ of $\Lambda^{\perp}_{\mathbf{R}}(\hat{\mathbf{a}})$ and an extension $\hat{\mathbf{c}}$ with $\mathbf{R}^{m_1} \ni \hat{\mathbf{a}} \sqsubset \hat{\mathbf{c}} \in \mathbf{R}^{m_1+m_2}$ as input. If $\hat{\mathbf{a}}$ generates $\mathbf{R}$, the algorithm outputs a basis $\mathbf{T}'$ for $\Lambda^{\perp}_{\mathbf{R}}(\hat{\mathbf{c}})$ with $\left\|\tilde{\mathbf{T}}'\right\| = \left\|\tilde{\mathbf{T}}\right\|$.*

The proposition is an adaptation of the respective proposition for q-ary lattices. The resulting trapdoor is

$$\mathbf{T}' = \left(\begin{array}{c|c} \mathbf{T} & \mathbf{V} \\ \hline \mathbf{0} & \mathbf{I}_{nm_2} \end{array}\right) \in \mathbb{Z}^{n(m_1+m_2) \times n(m_1+m_2)}.$$

Let $[\mathbf{v}_1, \ldots, \mathbf{v}_{nm_2}]$ be the columns of $\mathbf{V}$ with $\mathbf{v}_i \in \mathbb{Z}_q^{nm_1}$. Now, for every $i \in [nm_2]$, interpret $\mathbf{v}_i$ as an element $\hat{\mathbf{v}}_i \in \mathbf{R}_0^{m_1}$. They are chosen subject to $\hat{\mathbf{a}} \circledast \hat{\mathbf{v}}_i \equiv -\mathbf{b}_i$

but they need not be short. In consequence, $\mathbf{T}'$ is a basis of $\Lambda^{\perp}_{\mathbf{R}}(\hat{\mathbf{a}}\|\hat{\mathbf{b}})$. The Gram-Schmidt orthogonalization $\tilde{\mathbf{T}}'$ of $\mathbf{T}'$ has length $\left\|\tilde{\mathbf{T}}'\right\| = \left\|\tilde{\mathbf{T}}\right\|$ as $\mathbf{T}$ generates $\mathbb{Z}^{nm_1 \times nm_1}$ and, therefore,

$$\tilde{\mathbf{T}}' = \left(\begin{array}{c|c} \tilde{\mathbf{T}} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I}_{nm_2} \end{array}\right).$$

In this setting, RandBasis can be implemented exactly as in [CHKP10]; see Proposition 7.4.

8.2.2 Our Construction

We propose an identification scheme that is inspired by Lyubashevsky's construction in [Lyu08a, Lyu08b]. There, the prover has an associated public key $\mathbf{S} \in \mathbf{R}$ and a secret key $\hat{\mathbf{s}} \in \mathbf{R}_0^m$ such that $h(\hat{\mathbf{s}}) = \mathbf{S}$ and $\|\hat{\mathbf{s}}\|_\infty \leq 1$. The function h is chosen uniformly at random from $\mathcal{H}(\mathbf{R}, m)$ and it is the same for all users. Apart from the fact that it is identity-based, our 3-move identification scheme IBI scheme differs in three main aspects.

1. The user secrets $\hat{\mathbf{s}}$ have a larger norm and they are chosen from a different distribution;
2. All users share the *same* public value $\mathbf{S}$;
3. Each user proves knowledge under a *different* compression function h.

Informal Description We give a detailed, slightly informal description of the protocol Steps 1–4 in Figure 8.2 with the parameters in Table 8.1. Basically, the protocol provides a 3-move witness-indistinguishable proof of knowledge (cf. Section 2.5).

In the *first step*, the prover $\mathcal{P}$ picks random coins $\hat{\mathbf{y}} \leftarrow_{\$} D_y^m$ for this protocol run. Then, $\mathcal{P}$ commits to $\hat{\mathbf{y}}$ by sending $\mathbf{Y} = h_{\hat{\mathbf{a}}_{\mathsf{ID}}}(\hat{\mathbf{y}})$ to the verifier $\mathcal{V}$. The key $\hat{\mathbf{a}}_{\mathsf{ID}}$ to h is unique for each identity $\mathsf{ID} \in \{0,1\}^\lambda$ and it can be computed from the master public key $(\hat{\mathbf{a}}, \left\langle \hat{\mathbf{b}} \right\rangle, \mathbf{S})$.

In the *second step*, $\mathcal{V}$ challenges $\mathcal{P}$ with a random $\mathbf{c}$ from the set D_c.

The *third step* entails the computation of the response $\hat{\mathbf{z}}$ and checking whether it falls into a "safe" set G^m. If so, it is sent to $\mathcal{V}$. Otherwise, the protocol is restarted to ensure witness-indistinguishability. The abort ($\hat{\mathbf{z}} \leftarrow \perp$) happens with probability at most $1 - e^{-1/\phi}$ for our choice of D_y.

Parameter	Value	Asymptotics
m	$m_1 + (\lambda+1)m_2$	$\widetilde{\mathcal{O}}(\lambda)$
D_s	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq \tilde{L}\theta(nm_1 + nm_2) =: d_s\}$	$\widetilde{\mathcal{O}}(\sqrt{n})$
D_c	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq 1 =: d_c\}$	$\mathcal{O}(1)$
ϕ	positive integer constant ≥ 1	$\mathcal{O}(1)$
D_y	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq \phi mn^2 d_s =: d_y\}$	$\widetilde{\mathcal{O}}(n^2\sqrt{n}\lambda)$
G	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d_y - nd_s d_c =: d_G\}$	$\widetilde{\mathcal{O}}(n^2\sqrt{n}\lambda)$
D	$\{\mathbf{f} \in \mathbf{R}_0 : \|\mathbf{f}\|_\infty \leq d_G + nd_s d_c =: d_D\}$	$\widetilde{\mathcal{O}}(n^2\sqrt{n}\lambda)$
q (prime)	$\geq 4mn\sqrt{n}\log(n)d_D$	$\widetilde{\Theta}(\lambda n^4)$

The table defines all parameters for our scheme IBI. The parameters m_1, m_2, and θ are as per Proposition 8.2. The constant ϕ governs the completeness error and λ is the bit length of an identity. The third column contains the asymptotic growth for the respective norm bound or parameter with respect to λ and n.

Table 8.1: Parameters for the identity-based identification scheme IBI.

Finally, the verifier performs the actual verification in the *fourth step*. It involves testing that the coefficients of $\hat{\mathbf{z}}$ are within the correct interval and that the prover has used a correct secret key $\hat{\mathbf{s}}_{\mathsf{ID}}$, such that $h_{\hat{\mathbf{a}}_{\mathsf{ID}}}(\hat{\mathbf{s}}_{\mathsf{ID}}) = \mathbf{S}$, when computing $\hat{\mathbf{z}}$. This last check is possible due to the linearity of h.

Now, we explain how the secret key $\hat{\mathbf{s}}_{\mathsf{ID}}$ is extracted for a given identity ID. Let $\hat{\mathbf{a}}$ be the root of a bonsai tree and let $\left\langle \hat{\mathbf{b}} \right\rangle = \left((\hat{\mathbf{b}}_i^{(0)}, \hat{\mathbf{b}}_i^{(1)}) \right)_{i=1}^{\lambda}$ be the set of branches. Each identity $\mathsf{ID} = \mathsf{ID}_1 \| \ldots \| \mathsf{ID}_\lambda$ defines a unique path $\hat{\mathbf{a}}_{\mathsf{ID}} := \hat{\mathbf{a}} \| \hat{\mathbf{b}}_1^{(\mathsf{ID}_1)} \| \ldots \| \hat{\mathbf{b}}_\lambda^{(\mathsf{ID}_\lambda)}$ in the tree. Given a trapdoor $\mathbf{T}$ for the master lattice $\Lambda_{\mathbf{R}}^{\perp}(\hat{\mathbf{a}})$, we can find short vectors in the cosets $\{\hat{\mathbf{x}} : h_{\hat{\mathbf{a}}_{\mathsf{ID}}}(\hat{\mathbf{x}}) \equiv \mathbf{S}\}$ of any super lattice $\Lambda_{\mathbf{R}}^{\perp}(\hat{\mathbf{a}}_{\mathsf{ID}})$. The short elements in the cosets correspond to the per-identity secret keys $\hat{\mathbf{s}}_{\mathsf{ID}}$.

In the proof, we have to set up the tree such that it comprises both controlled and uncontrolled paths. To do that, it is important to know all queries to the extraction oracle before handing over the master public key; hence, the weak security model. Since the attacker will impersonate an identity that it has not queried to $\mathsf{OExtract}$, the attacked identity is likely to overlap only with branches of uncontrolled growth.

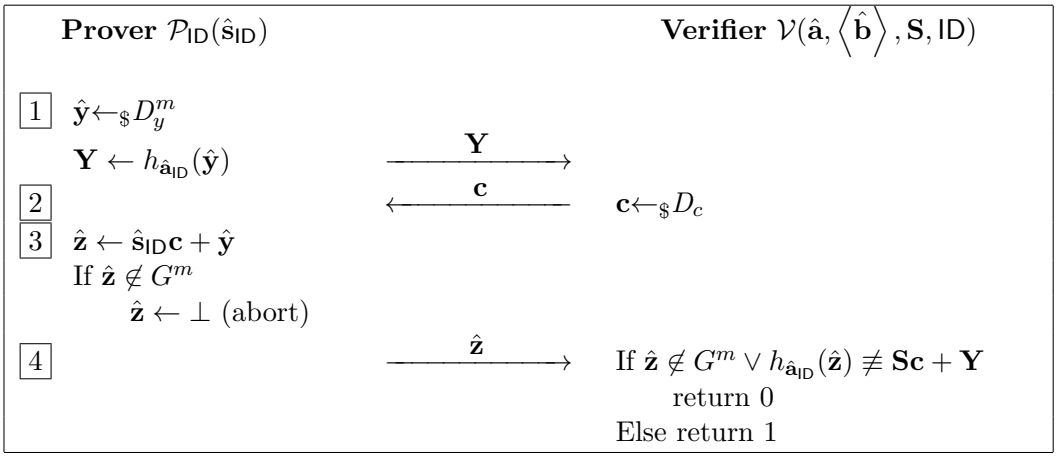


Figure 8.2: Identity-based identification protocol.

There, the simulator embeds the challenge from the collision problem.

Maybe the most interesting part of the construction is the simulation technique for arbitrary provers. It uses a single, universal secret key $\hat{\mathbf{s}}^\star \in \mathbf{R}_0^{m_1+m_2}$, such that $h_{\hat{\mathbf{a}}}(\hat{\mathbf{s}}^\star) = \mathbf{S}$, for all provers. This allows a very efficient simulation. In particular, it is not necessary to use the, rather inefficient, $\mathsf{Extract}$ algorithm to set up a prover. The witness indistinguishability hides this deviation from the real protocol. Now, let us define the scheme formally.

Master-key Generation: Kg performs the following steps.

- $\hat{\mathbf{a}}_0 \leftarrow_{\$} \mathbf{R}^{m_1}$;
- $(\hat{\mathbf{a}}, \mathbf{T}) \leftarrow \mathsf{ExtLattice}(\hat{\mathbf{a}}_0, m_1 + m_2)$;
- $\langle \hat{\mathbf{b}} \rangle := \left((\hat{\mathbf{b}}_i^{(0)}, \hat{\mathbf{b}}_i^{(1)}) \right)_1^\lambda$ with $\hat{\mathbf{b}}_i^{(b)} \leftarrow_{\$} \mathbf{R}^{m_2}$ for all $i \in [\lambda]$ and $b \in \{0,1\}$;
- $\mathbf{S} \leftarrow_{\$} \mathbf{R}$.

Finally, it outputs the master secret key $\mathbf{T}$ and the master public key $(\hat{\mathbf{a}}, \langle \hat{\mathbf{b}} \rangle, \mathbf{S})$.

Key Extraction: $\mathsf{Extract}(\mathbf{T}, \mathsf{ID})$ parses $\mathsf{ID} = \mathsf{ID}_1 \| \cdots \| \mathsf{ID}_\lambda$ and sets $\hat{\mathbf{a}}_{\mathsf{ID}} := \hat{\mathbf{a}} \| \hat{\mathbf{b}}_1^{(\mathsf{ID}_1)} \| \cdots \| \hat{\mathbf{b}}_\lambda^{(\mathsf{ID}_\lambda)} \in \mathbf{R}^m$. Then, it samples $\hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_\lambda$ via $\mathsf{SampleDom}(m_2, \eta)$ and calls $\hat{\mathbf{s}}_0 \leftarrow \mathsf{SamplePre}(\mathbf{T}, \eta, \mathbf{S} - \sum_{i=1}^\lambda \hat{\mathbf{b}}_i^{(\mathsf{ID}_i)} \circledast \hat{\mathbf{s}}_i \bmod q)$.

The output is $\hat{\mathbf{s}}_{\mathsf{ID}} \leftarrow \hat{\mathbf{s}}_0 \| \cdots \| \hat{\mathbf{s}}_\lambda \in D_s^m$. In the event that $\hat{\mathbf{s}}_{\mathsf{ID}} \notin D_s^m$, the algorithm starts over.

Identification Protocol: See Figure 8.2. When the protocol aborts, the prover starts over with a fresh $\hat{\mathbf{y}}$.

Notice that our scheme can be also be adapted to support a hierarchy of identities, each acting as the key extraction authority for its subordinates. Thus, each user receives a secret key, a trapdoor for a super lattice, that can be used to generate the secret key for the identification scheme. This adaptation involves adding more layers to the bonsai tree and applying RandBasis during basis delegation to prevent leaking information about the master trapdoor.

The analysis of the scheme heavily relies on the results for our blind signature scheme in Chapter 4. Both schemes share a common structure and a common set of techniques. We have presented the lemmas in Section 4.2 in a generic way so as to also cover the setting here.

Theorem 8.4 (Completeness)**.** *Let $g(n) = \omega(\log(n))$. The scheme IBI is complete after at most $g(n)$ (or, an expected number of $e^{1/\phi}$) restarts.*

Proof. For all honestly generated master-key pairs $(\mathbf{T}, (\hat{\mathbf{a}}, \left\langle \hat{\mathbf{b}} \right\rangle, \mathbf{S}))$, and all identities $\mathsf{ID} = \mathsf{ID}_1 \| \cdots \| \mathsf{ID}_\lambda$, the key extraction algorithm outputs a secret key $\hat{\mathbf{s}}_{\mathsf{ID}} = \hat{\mathbf{s}}_0 \| \ldots \| \hat{\mathbf{s}}_\lambda \in D_s^m$ with $h_{\hat{\mathbf{a}}_{\mathsf{ID}}}(\hat{\mathbf{s}}_{\mathsf{ID}}) \equiv h_{\hat{\mathbf{a}}}(\hat{\mathbf{s}}_0) + \sum_{i=1}^{\lambda} h_{\hat{\mathbf{b}}_i^{(\mathsf{ID}_i)}}(\hat{\mathbf{s}}_i) \equiv \mathbf{S} - \sum_{i=1}^{\lambda} h_{\hat{\mathbf{b}}_i^{(\mathsf{ID}_i)}}(\hat{\mathbf{s}}_i)$ $+ \sum_{i=1}^{\lambda} h_{\hat{\mathbf{b}}_i^{(\mathsf{ID}_i)}}(\hat{\mathbf{s}}_i) \equiv \mathbf{S}$ and $\|\hat{\mathbf{s}}_{\mathsf{ID}}\|_\infty \leq d_s$.

For all challenges $\mathbf{c} \in D_c$ and all random coins $\hat{\mathbf{y}} \in D_y^m$, we have $\|\hat{\mathbf{z}}\|_\infty = \|\hat{\mathbf{s}}_{\mathsf{ID}}\mathbf{c} + \hat{\mathbf{y}}\|_\infty \leq d_y - n \|\hat{\mathbf{s}}_{\mathsf{ID}}\|_\infty \|c\|_\infty = d_y - n = d_G$ with probability at least $e^{-1/\phi} - o(1)$ because of Lemma 4.2 and Lemma 4.1 (with $k = mn, A = nd_s d_c, B = d_y$). Hence, the verifier accepts because $h_{\hat{\mathbf{a}}_{\mathsf{ID}}}(\hat{\mathbf{z}}) \equiv h_{\hat{\mathbf{a}}_{\mathsf{ID}}}(\hat{\mathbf{s}}_{\mathsf{ID}})\mathbf{c} + h_{\hat{\mathbf{a}}_{\mathsf{ID}}}(\hat{\mathbf{y}}) \equiv \mathbf{S}\mathbf{c} + \mathbf{Y}$.

Repeating the protocol $\omega(\log(n))$ times in parallel establishes completeness. In practice, a small and constant number $e^{1/\phi}$ of retries is sufficient. □

Observe that in any case, all operations (including eventual restarts) in IBI.Protocol have $\widetilde{\mathcal{O}}(n\lambda)$ complexity and that private keys, public keys, protocol messages, as well as the master public key have size $\widetilde{\mathcal{O}}(n\lambda)$. The only exceptions are the master secret key size, which has $\widetilde{\mathcal{O}}(n^2\lambda)$ bits, and the key extraction algorithm Extract. Fortunately, the online phase merely requires quasi-linear operations and a quasi-linear bandwidth in n. However, when applying Theorem 8.1, we require a large identity space, e.g., $\lambda = n$.

8.2.3 Security

Since the function family $\mathcal{H}(\mathbf{R}, k)$ compresses the domain D_s^k for $k \geq m_1 + m_2$, it is easy to show that all secret keys collide with at least one other secret key (cf. Lemma 4.6). We prove soundness of the protocol in the following theorem. Furthermore, the protocol can be shown to be witness indistinguishable with respect to the secret per-identity keys (cf. Lemma 4.7). Thus, we can securely use parallel composition.

Theorem 8.5 (Soundness)**.** *IBI is secure in the STAT-ID-IMP-CA model if the collision problem* $COL(\mathcal{H}(\mathbf{R}, m + \lambda m_2), D)$ *is hard.*

The main idea is to exploit witness indistinguishability and simulate all provers with a single secret key $\hat{\mathbf{s}}^\star = \hat{\mathbf{s}}_0^\star \| \hat{\mathbf{0}} \| \ldots \| \hat{\mathbf{0}} \in \mathbf{R}_0^m$, where $\hat{\mathbf{s}}_0^\star \in \mathbf{R}_0^{m_1+m_2}$, $\|\hat{\mathbf{s}}_0^\star\|_\infty \leq \eta\theta(m_1 + m_2)$, and $\hat{\mathbf{a}} \circledast \hat{\mathbf{s}}_0^\star \equiv \mathbf{S}$, which is prepared during setup of the simulation.

Moreover, we prepare the set $\left\langle \hat{\mathbf{b}} \right\rangle$ so that we know a trapdoor for certain branches of the tree, while others are embedded with the external challenge from the collision problem. With good probability, the adversary will impersonate an identity that corresponds to such uncontrolled branches.

During this phase of the attack, we run the knowledge extractor of the underlying proof of knowledge to obtain a secret key $\hat{\mathbf{s}}' \neq \hat{\mathbf{s}}^\star$ and solve the collision problem. This step involves the Reset Lemma [BP02], which is repeated as Lemma 8.7 at the end of this chapter in Section 8.2.4.

Proof. The following reduction algorithm $\mathcal{B}^{\mathcal{A}}$ attempts to solve the collision problem and has access to an efficient, successful adversary $\mathcal{A}$ against IBI in the STAT-ID-IMP-CA experiment.

Setup: $\mathcal{B}$ receives $\hat{\mathbf{c}} = \hat{\mathbf{a}} \| \hat{\mathbf{u}}_1^{(0)} \| \hat{\mathbf{u}}_1^{(1)} \| \cdots \| \hat{\mathbf{u}}_\lambda^{(0)} \| \hat{\mathbf{u}}_\lambda^{(1)} \in \mathbf{R}^{m_1+m_2} \| \mathbf{R}^{2\lambda m_2}$. It invokes $\mathcal{A}(m\_find)$ to obtain *distinct* $\mathsf{ID}_1, \ldots, \mathsf{ID}_Q \in \{0,1\}^\lambda$. Let $\langle \pi \rangle := (\pi_i)_1^p$ be the set of all strings $\pi \in \{0,1\}^\lambda$ such that $\pi \not\sqsubset \mathsf{ID}_j$ for $j \in [Q]$ and $\pi_i \not\sqsubset \pi_j$ for all distinct pairs (i, j) in $\langle \pi \rangle$. The set $\langle \pi \rangle$ contains at most $\lambda \cdot Q$ elements. $\mathcal{B}$ randomly selects $\pi \leftarrow_\$ \langle \pi \rangle$, which represents the challenge subtree. Let $|\pi| = l_\pi$. The public key is set up as follows.

- $\hat{\mathbf{b}}_i^{(\pi_i)} \leftarrow \hat{\mathbf{u}}_i^{(0)}$ for $i = 1, \ldots, l_\pi$;
- $\hat{\mathbf{b}}_i^{(b)} \leftarrow \hat{\mathbf{u}}_i^{(b)}$ for $b \in \{0,1\}$ and $i = l_\pi + 1, \ldots, \lambda$;
- $\hat{\mathbf{b}}_i^{1-\pi_i}$ and $\mathbf{T}_i$ via $\mathsf{ExtLattice}(\hat{\mathbf{a}}^\star \| \hat{\mathbf{b}}_1^{(\pi_1)} \| \cdots \| \hat{\mathbf{b}}_{i-1}^{(\pi_{i-1})}, m_2)$ for $i = 1, \ldots, l_\pi$.

Then, $\mathcal{B}$ uses $\mathsf{SampleDom}$ to sample an element $\hat{\mathbf{s}}_0^\star \in \mathbf{R}_0^{m_1+m_2}$ and computes $\mathbf{S} \leftarrow h_{\hat{\mathbf{a}}}(\hat{\mathbf{s}}_0^\star)$. The universal simulation key will be $\hat{\mathbf{s}}^\star \leftarrow \hat{\mathbf{s}}_0^\star \| \hat{\mathbf{0}} \| \cdots \| \hat{\mathbf{0}} \in \mathbf{R}_0^m$. For

each identity $\mathsf{ID}_i = I_1^{(i)} \| \cdots \| I_\lambda^{(i)}$, let j be the smallest index with $I_j^{(i)} \neq \pi_j$. $\mathcal{B}$ computes the secret key $\hat{\mathbf{s}}_{\mathsf{ID}_i} \leftarrow \mathsf{SamplePre}(\mathsf{ExtBasis}(\mathbf{T}_j, \hat{\mathbf{a}}_{\mathsf{ID}_i}), \eta, \mathbf{S})$. The public key comprises $\hat{\mathbf{a}}$, $\mathbf{S}$, and $\left\langle \hat{\mathbf{b}} \right\rangle := \left((\hat{\mathbf{b}}_i^{(0)}, \hat{\mathbf{b}}_i^{(1)}) \right)_{i=1}^{\lambda}$ and the reduction returns the public key and the list of secret keys to $\mathcal{A}$.

Prover Queries: $\mathcal{A}$ may challenge $\mathcal{B}$ with any identity ID. The simulator acts as per the protocol in Figure 8.2 but uses the same secret $\hat{\mathbf{s}}^\star$ for all identities.

Impersonation Attempt: At some point, $\mathcal{A}$ outputs a challenge identity ID^*, which has not been queried to the extraction oracle before. $\mathcal{B}$ challenges $\mathcal{A}$ to prove knowledge of a secret key for ID^*. After receiving the commitment $\mathbf{Y}$, $\mathcal{B}$ sends $\mathbf{c_1} \leftarrow_\$ D_c$, and receives $\hat{\mathbf{z}}_1$. Then, the reduction rewinds $\mathcal{A}$ to the end of Step 1 and challenges the adversary with $\mathbf{c_2} \leftarrow_\$ D_c$ to obtain the answer $\hat{\mathbf{z}}_2$. The reduction suitably rearranges and pads (with $\hat{\mathbf{0}}$) the pair $(\hat{\mathbf{z}}_1 - \hat{\mathbf{s}}\mathbf{c_1}, \hat{\mathbf{z}}_1 - \hat{\mathbf{s}}\mathbf{c_2})$ and outputs the result as its solution to the problem COL under $\hat{\mathbf{c}}$.

Analysis First of all, observe that *mpk* in the simulation is statistically indistinguishable from *mpk* in the real scheme (cf. Proposition 8.2). Furthermore, note that the simulator can answer all extraction queries correctly because it knows a trapdoor for a prefix of all requested identities. The prover queries are answered correctly as $\hat{\mathbf{s}}^\star$ satisfies the relation $h_{\mathsf{ID}}(\hat{\mathbf{s}}^\star) = \mathbf{S}$ for *any* ID.

Let us assume that the reset during $\mathcal{A}$'s impersonation attempt immediately yields another valid response. Then, we certainly have $h_{\hat{\mathbf{a}}_{\mathsf{ID}^*}}(\hat{\mathbf{z}}_1 - \hat{\mathbf{s}}^\star\mathbf{c_1}) = \mathbf{Y} = h_{\hat{\mathbf{a}}_{\mathsf{ID}^*}}(\hat{\mathbf{z}}_2 - \hat{\mathbf{s}}^\star\mathbf{c_2})$ with $\max\{\|\hat{\mathbf{z}}_1 - \hat{\mathbf{s}}^\star\mathbf{c_1}\|_\infty, \|\hat{\mathbf{z}}_2 - \hat{\mathbf{s}}^\star\mathbf{c_2}\|_\infty\} \leq d_G + nd_sd_c = d_D$. What is left to show is that $\hat{\mathbf{z}}_1 - \hat{\mathbf{s}}^\star\mathbf{c_1} \neq \hat{\mathbf{z}}_2 - \hat{\mathbf{s}}^\star\mathbf{c_2}$. Lemma 4.6 guarantees the existence of at least two distinct valid secret keys $\hat{\mathbf{s}}^\star$ and $\hat{\mathbf{s}}'$. Now, for one of them, we obtain a valid collision. Assuming the contrary, $\hat{\mathbf{z}}_1 - \hat{\mathbf{s}}^\star\mathbf{c_1} = \hat{\mathbf{z}}_2 - \hat{\mathbf{s}}^\star\mathbf{c_2}$ and $\hat{\mathbf{z}}_1 - \hat{\mathbf{s}}'\mathbf{c_1} = \hat{\mathbf{z}}_2 - \hat{\mathbf{s}}'\mathbf{c_2}$ yields $\mathbf{c_1}(\hat{\mathbf{s}}' - \hat{\mathbf{s}}^\star) = \mathbf{c_2}(\hat{\mathbf{s}}' - \hat{\mathbf{s}}^\star)$ and therefore $(\mathbf{c_1} - \mathbf{c_2})(\hat{\mathbf{s}}' - \hat{\mathbf{s}}^\star) = \hat{\mathbf{0}}$. This only holds if $\hat{\mathbf{s}}' = \hat{\mathbf{s}}^\star$ because $\max\{\|\hat{\mathbf{s}}^\star\|_\infty, \|\hat{\mathbf{s}}'\|_\infty\} \leq q/2$ and $\mathbf{R}_0$ is an integral domain. Thus, with probability $\geq 1/2$, the simulator can use $\mathcal{A}$'s output to solve COL.

Concerning the success probability of the reset, assume that $\mathcal{A}$ is successful with non-negligible probability ϵ. Then, $\mathcal{A}$ is successful with non-negligible probability $\geq (\epsilon - 1/|D_c|)^2$ due to the Reset Lemma.

All in all, the success probability of the simulator against the collision problems stays non-negligible if ϵ is non-negligible. □

Proposition 2.1 yields the following corollary and Theorem 8.1 yields an identity-based identification scheme that is secure in the ADAPT-ID-IMP-CA model.

Corollary 8.6. *IBI is secure in the STAT-ID-IMP-CA model if SIVP$^\infty$ is hard in the worst case for approximation factors $\gamma = \widetilde{\mathcal{O}}(\lambda n^3\sqrt{n})$ in lattices that correspond to ideals in $\mathbf{R}_0$.*

8.2.4 Reset Lemma

For the readers convenience, we formulate it in terms of our identification scheme directly. In fact, it is more general.

Lemma 8.7 (Reset Lemma [BP02])**.** *Let $\mathcal{P}$ be prover in IBI and let $\mathcal{V}$ be the verifier. The prover is split into two algorithms $\mathcal{P}_1$ and $\mathcal{P}_3$, corresponding to Step 1 and Step 3 in the protocol. The verifier consists of the challenge set D_c and a verification algorithm Vf (Step 4). With s_prover, we denote the state of $\mathcal{P}$. Let mpk be a master public key, generated by Kg and let ID be the identity of $\mathcal{P}$. The probability that $\mathcal{V}$ accepts when interacting with $\mathcal{P}$ is*

$$\mathsf{acc} = \mathrm{Prob}\left[d = 1 \;\middle|\; \begin{array}{l} (\mathbf{Y}, s\_prover) \leftarrow_\$ \mathcal{P}_1(mpk, \mathsf{ID}); \\ \mathbf{c} \leftarrow_\$ D_c; \\ \hat{\mathbf{z}} \leftarrow_\$ \mathcal{P}_3(\mathbf{c}, s\_prover); \\ d \leftarrow \mathsf{Vf}(mpk, \mathsf{ID}, \mathbf{Y}, \mathbf{c}, \hat{\mathbf{z}}); \end{array} \right].$$

Then, the following probability, where the $\mathcal{P}$gets two challenges $\mathbf{c}$ for the same commitment $\mathbf{Y}$, is

$$\mathsf{res} = \mathrm{Prob}\left[d_1 = d_2 = 1 \wedge \mathbf{c}_1 \neq \mathbf{c}_2 : \begin{array}{l} (\mathbf{Y}, s\_prover) \leftarrow_\$ \mathcal{P}_1(mpk, \mathsf{ID}); \\ \mathbf{c}_1 \leftarrow_\$ D_c; \\ \hat{\mathbf{z}} \leftarrow_\$ \mathcal{P}_3(\mathbf{c}_1, s\_prover); \\ d_1 \leftarrow \mathsf{Vf}(mpk, \mathsf{ID}, \mathbf{Y}, \mathbf{c}_1, \hat{\mathbf{z}}); \\ \mathbf{c}_2 \leftarrow_\$ D_c; \\ \hat{\mathbf{z}} \leftarrow_\$ \mathcal{P}_3(\mathbf{c}_2, s\_prover); \\ d_2 \leftarrow \mathsf{Vf}(mpk, \mathsf{ID}, \mathbf{Y}, \mathbf{c}_2, \hat{\mathbf{z}}); \end{array} \right]$$

with $\mathsf{res} \geq (\mathsf{acc} - 1/|D_c|)^2$.

8.3 Conclusion and Open Problems

Using a new, weaker security model for identity-based identification and a generic transformation toward full security, we have shown how to construct an identity-based identification scheme from lattices that is secure against concurrent impersonation and adaptive-identity attacks in the standard model. Our scheme is asymptotically efficient and at its core, we use a new efficient simulation technique for bonsai

tree constructions. In practice, the most efficient alternative would be using the certification approach [BNN09] with [CLRS10], for which the dissertation author has contributed the security proofs.

One of the main open problems is to improve the efficiency of such direct constructions and, if possible, avoid trapdoor bases entirely.

Bibliography

[AB09] Elli Androulaki and Steven M. Bellovin. An anonymous credit card system. In Simone Fischer-Hübner, Costas Lambrinoudakis, and Günther Pernul, editors, *TrustBus*, volume 5695 of *Lecture Notes in Computer Science*, pages 42–51. Springer, 2009.

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (h)ibe in the standard model. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572. Springer, 2010.

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical ibe. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 98–115. Springer, 2010.

[Abe01] Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 136–151. Springer, 2001.

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC*, pages 284–293, 1997.

[AD07] Miklós Ajtai and Cynthia Dwork. The first and fourth public-key cryptosystems with worst-case/average-case equivalence. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(097), 2007.

[ADL+08] Y. Arbitman, G. Dogon, V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFTX: A proposal for the SHA-3 standard, 2008. In the First SHA-3 Candidate Conference.

[ADR02] Jee Hea An, Yevgeniy Dodis, and Tal 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.

[ADW09] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 36–54. Springer, 2009.

[AF96] Masayuki Abe and Eiichiro Fujisaki. How to date blind signatures. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT*, volume 1163 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1996.

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108. ACM, 1996.

[Ajt99] Miklós Ajtai. Generating hard instances of the short basis problem. In Jirí Wiedermann, Peter van Emde Boas, and Mogens Nielsen, editors, *ICALP*, volume 1644 of *Lecture Notes in Computer Science*, pages 1–9. Springer, 1999.

[ANN06] Michel Abdalla, Chanathip Namprempre, and Gregory Neven. On the (im)possibility of blind message authentication codes. In David Pointcheval, editor, *CT-RSA*, volume 3860 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2006.

[AO09] Masayuki Abe and Miyako Ohkubo. A framework for universally composable non-committing blind signatures. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 435–450. Springer, 2009.

[AP08] Joel Alwen and Chris Peikert. Generating shorter bases for hard random lattices. Cryptology ePrint Archive, Report 2008/521, 2008. `http://eprint.iacr.org/`.

[AP09] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. In Susanne Albers and Jean-Yves Marion, editors, *STACS*, volume 09001 of *Dagstuhl Seminar Proceedings*, pages 75–86. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2009.

[ASW00] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18(4):593–610, 2000.

[Bab86] László Babai. On lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[BBC+94] Jean-Paul Boly, Antoon Bosselaers, Ronald Cramer, Rolf Michelsen, Stig Fr. Mjølsnes, Frank Muller, Torben P. Pedersen, Birgit Pfitzmann, Peter de Rooij, Berry Schoenmakers, Matthias Schunter, Luc Vallée, and Michael Waidner. The esprit project cafe - high security digital payment systems. In Dieter Gollmann, editor, *ESORICS*, volume 875 of *Lecture Notes in Computer Science*, pages 217–230. Springer, 1994.

[BBD08] Daniel J. Bernstein, Johannes A. Buchmann, and Erik Dahmen, editors. *Post-Quantum Cryptography*. Springer, 2008.

[BCHK07] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM J. Comput.*, 36(5):1301–1328, 2007.

[BDM98] Feng Bao, Robert H. Deng, and Wenbo Mao. Efficient and practical fair exchange protocols with off-line ttp. In *IEEE Symposium on Security and Privacy*, pages 77–85. IEEE Computer Society, 1998.

[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. In Hugo Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer, 1998.

[BDR+96] Matt Blaze, Whitfield Diffie, Ronald L. Rivest, Bruce Schneier, Tsutomu Shimomura, Eric Thompson, and Michael Wiener. Minimal key

lengths for symmetric ciphers to provide adequate commercial security. A Report by an Ad Hoc Group of Cryptographers and Computer Scientists, 1996.

[BDS08] Johannes Buchmann, Erik Dahmen, and Michael Szydlo. Hash-based cryptography. In Bernstein et al. [BBD08], pages 35–93.

[Ber09] Daniel J. Bernstein. Cost analysis of hash collisions: Will quantum computers make sharcs obsolete? Workshop Record of SHARCS'09: Special-purpose Hardware for Attacking Cryptographic Systems, 2009.

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In Ernest F. Brickell, editor, *CRYPTO*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer, 1992.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer, 2003.

[BGR98a] Mihir Bellare, Juan A. Garay, and Tal Rabin. Batch verification with applications to cryptography and checking. In Claudio L. Lucchesi and Arnaldo V. Moura, editors, *LATIN*, volume 1380 of *Lecture Notes in Computer Science*, pages 170–191. Springer, 1998.

[BGR98b] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In *EUROCRYPT*, pages 236–250, 1998.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum cryptanalysis of hash and claw-free functions. In Claudio L. Lucchesi and Arnaldo V. Moura, editors, *LATIN*, volume 1380 of *Lecture Notes in Computer Science*, pages 163–169. Springer, 1998.

[BKM09] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology*, 22(1):114–138, 2009.

[BLR08] Johannes Buchmann, Richard Lindner, and Markus Rückert. Explicit hard instances of the shortest vector problem. In Johannes Buchmann

and Jintai Ding, editors, *PQCrypto*, volume 5299 of *Lecture Notes in Computer Science*, pages 79–94. Springer, 2008.

[BLRS08] Johannes Buchmann, Richard Lindner, Markus Rückert, and Michael Schneider. Explicit hard instances of the shortest vector problem (extended version). Cryptology ePrint Archive, Report 2008/333, 2008. `http://eprint.iacr.org/`.

[BLRS09] Johannes Buchmann, Richard Lindner, Markus Rückert, and Michael Schneider. Post-quantum cryptography: lattice signatures. *Computing*, 85(1-2):105–125, 2009.

[BMV08] Emmanuel Bresson, Jean Monnerat, and Damien Vergnaud. Separation results on the "one-more" computational problems. In Tal Malkin, editor, *CT-RSA*, volume 4964 of *Lecture Notes in Computer Science*, pages 71–87. Springer, 2008.

[BMW03] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629. Springer, 2003.

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security*, pages 390–399. ACM, 2006.

[BNN09] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Security proofs for identity-based identification and signature schemes. *J. Cryptology*, 22(1):1–61, 2009.

[BNPS03] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The one-more-rsa-inversion problems and the security of chaum's blind signature scheme. *J. Cryptology*, 16(3):185–215, 2003.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In Yvo Desmedt, editor, *Public Key Cryptography*, volume

2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2003.

[Boy10] Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In Phong Q. Nguyen and David Pointcheval, editors, *Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 499–517. Springer, 2010.

[BP02] Mihir Bellare and Adriana Palacio. Gq and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 162–177. Springer, 2002.

[BP10] Stefan Brands and Christian Paquin. U-prove cryptographic specification v1.0. `http://connect.microsoft.com/site642/Downloads/DownloadDetails.aspx?DownloadID=26953`, March 2010.

[BR93] Mihir Bellare and Pil Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS*. ACM, 1993.

[BR96] Mihir Bellare and Phillip 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.

[Bra99] Stefan Brands. *Rethinking Public Key Infrastructure and Digital Certificates— Building in Privacy.* PhD thesis, Eindhoven Institute of Technology, 1999.

[BS07] Mihir Bellare and Sarah Shoup. Two-tier signatures, strongly unforgeable signatures, and fiat-shamir without random oracles. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 201–216. Springer, 2007.

[BSW06] Dan Boneh, Emily Shen, and Brent Waters. Strongly unforgeable signatures based on computational diffie-hellman. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 229–240. Springer, 2006.

[CG08] Jan Camenisch and Thomas Groß. Efficient attributes for anonymous credentials. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security*, pages 345–356. ACM, 2008.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004.

[Cha82] David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, pages 199–203, 1982.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552. Springer, 2010.

[CHP07] Jan Camenisch, Susan Hohenberger, and Michael Østergaard Pedersen. Batch verification of short signatures. In Moni Naor, editor, *EUROCRYPT*, volume 4515 of *Lecture Notes in Computer Science*, pages 246–263. Springer, 2007.

[CKW04] Jan Camenisch, Maciej Koprowski, and Bogdan Warinschi. Efficient blind signatures without random oracles. In *Security in Communication Networks*, volume 3352 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 2004.

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer, 2001.

[CLRS10] Pierre-Louis Cayrel, Richard Lindner, Markus Rückert, and Rosemberg Silva. Improved zero-knowledge identification with lattices. In Swee-Huay Henc and Kaoru Kurosawa, editors, *ProvSec*, volume 6402 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2010.

[CNR10] Jan Camenisch, Gregory Neven, and Markus Rückert. Lattice-based group signatures from anonymous attribute tokens. Manuscript, 2010.

[CNS07] Jan Camenisch, Gregory Neven, and Abhi Shelat. Simulatable adaptive oblivious transfer. In Moni Naor, editor, *EUROCRYPT*, volume

4515 of *Lecture Notes in Computer Science*, pages 573–590. Springer, 2007.

[CSF+08] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard), May 2008.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.

[DFLS10] Özgür Dagdelen, Marc Fischlin, Anja Lehmann, and Christian Schaffner. Random oracles in a quantum world. Cryptology ePrint Archive, Report 2010/428, 2010. `http://eprint.iacr.org/`.

[DLY07] Yevgeniy Dodis, Pil Joong Lee, and Dae Hyun Yum. Optimistic fair exchange in a multi-user setting. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 118–133. Springer, 2007.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302. IEEE Computer Society, 2008.

[ECR10] ECRYPT2. Yearly report on algorithms and keysizes — report D.SPA.13, 2010. available at `http://www.ecrypt.eu.org/documents/D.SPA.13.pdf`.

[EGM96] Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. *J. Cryptology*, 9(1):35–67, 1996.

[Fis06] Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 60–77. Springer, 2006.

[FP96] Christopher A. Fuchs and Asher Peres. Quantum-state disturbance versus information gain: Uncertainty relations for quantum information. *Phys. Rev. A*, 53(4):2038–2045, Apr 1996.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko,

editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *STOC*, pages 416–426. ACM, 1990.

[FS09] Marc Fischlin and Dominique Schröder. Security of blind signatures under aborts. In Stanislaw Jarecki and Gene Tsudik, editors, *Public Key Cryptography*, volume 5443 of *Lecture Notes in Computer Science*, pages 297–316. Springer, 2009.

[GGH96] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Collision-free hashing from lattice problems. *Electronic Colloquium on Computational Complexity (ECCC)*, 3(42), 1996.

[GHGKN06] Nicolas Gama, Nick Howgrave-Graham, Henrik Koy, and Phong Q. Nguyen. Rankin's constant and blockwise lattice reduction. In *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 112–130. Springer, 2006.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. A simple bgn-type cryptosystem from lwe. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2010.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GM03] Daniel Goldstein and Andrew Mayer. On the equidistribution of Hecke points. *Forum Mathematicum 2003, 15:2*, pages 165–189, 2003.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[GN08a] Nicolas Gama and Phong Q. Nguyen. Finding short lattice vectors within Mordell's inequality. In *STOC*, pages 207–216. ACM, 2008.

[GN08b] Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 31–51. Springer, 2008.

[GNR10] Nicolas Gama, Phong Q. Nguyen, and Oded Regev. Lattice enumeration using extreme pruning. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 257–278. Springer, 2010.

[Gol04] Oded Goldreich. *The Foundations of Cryptography*, volume 1. Cambridge University Press, 2004.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *STOC*, pages 197–206. ACM, 2008.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *STOC*, pages 212–219. ACM, 1996.

[GS02] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In Yuliang Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[HKKL07] Carmit Hazay, Jonathan Katz, Chiu-Yuen Koo, and Yehuda Lindell. Concurrently-secure blind signatures without random oracles or setup assumptions. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 323–341. Springer, 2007.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. Ntru: A ring-based public key cryptosystem. In Joe Buhler, editor, *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998.

[HW09] Susan Hohenberger and Brent Waters. Short and stateless signatures from the rsa assumption. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 654–670. Springer, 2009.

[IKSA03] S. Ibrahim, M. Kamat, M. Salleh, and S.R.A. Aziz. Secure e-voting with blind signature. In *Telecommunication Technology, 2003. NCTT*

2003 Proceedings. 4th National Conference on, pages 193 – 197, January 2003.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *Structure in Complexity Theory Conference*, pages 134–147, 1995.

[JLO97] Ari Juels, Michael Luby, and Rafail Ostrovsky. Security of blind digital signatures (extended abstract). In Burton S. Jr. Kaliski, editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 150–164. Springer, 1997.

[Kan83] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *STOC*, pages 193–206. ACM, 1983.

[KH05] Kaoru Kurosawa and Swee-Huay Heng. Identity-based identification without random oracles. In Osvaldo Gervasi, Marina L. Gavrilova, Vipin Kumar, Antonio Laganà, Heow Pueh Lee, Youngsong Mun, David Taniar, and Chih Jeng Kenneth Tan, editors, *ICCSA (2)*, volume 3481 of *Lecture Notes in Computer Science*, pages 603–613. Springer, 2005.

[Kle00] Philip N. Klein. Finding the closest lattice vector when it's unusually close. In *SODA*, pages 937–941, 2000.

[KR00] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS*. The Internet Society, 2000.

[KTX07] Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Multi-bit cryptosystems based on lattice problems. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 315–329. Springer, 2007.

[KTX08] Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In Josef Pieprzyk, editor, *ASIACRYPT*, volume 5350 of *Lecture Notes in Computer Science*, pages 372–389. Springer, 2008.

[KV09] Jonathan Katz and Vinod Vaikuntanathan. Signature schemes with bounded leakage resilience. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 703–720. Springer, 2009.

[Len05] Arjen Lenstra. *The Handbook of Information Security*, chapter 114 — Key Lengths. Wiley, 2005. available at `http://www.keylength.com/biblio/Handbook_of_Information_Security_-_Keylength.pdf`.

[LM06] Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 144–155. Springer, 2006.

[LM08] Vadim Lyubashevsky and Daniele Micciancio. Asymptotically efficient lattice-based digital signatures. In Ran Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 37–54. Springer, 2008.

[LMPR08] Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. Swifft: A modest proposal for fft hashing. In Kaisa Nyberg, editor, *FSE*, volume 5086 of *Lecture Notes in Computer Science*, pages 54–72. Springer, 2008.

[LOS+06] Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 465–485. Springer, 2006.

[LP10] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for lwe-based encryption. Cryptology ePrint Archive, Report 2010/613, 2010. `http://eprint.iacr.org/`.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.

[LV01] Arjen Lenstra and Eric R. Verheul. Selecting cryptographic key sizes. *J. Cryptology*, 14(4):255–293, 2001.

[Lyu08a] Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In Ronald Cramer, editor, *Public Key Cryptogra-*

phy, volume 4939 of *Lecture Notes in Computer Science*, pages 162–179. Springer, 2008.

[Lyu08b] Vadim Lyubashevsky. *Towards Practical Lattice-Based Cryptography.* PhD thesis, University of California, San Diego, 2008.

[Lyu09] Vadim Lyubashevsky. Fiat-shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616. Springer, 2009.

[Mer89] Ralph C. Merkle. A certified digital signature. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer, 1989.

[Mic01] Daniele Micciancio. Improving lattice based cryptosystems using the hermite normal form. In Joseph H. Silverman, editor, *CaLC*, volume 2146 of *Lecture Notes in Computer Science*, pages 126–145. Springer, 2001.

[Mic07] Daniele Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411, 2007.

[Mic10] Microsoft Corporation. Creating certificate policies and certificate practice statements. `http://technet.microsoft.com/en-us/library/cc780454(WS.10).aspx`, February 2010.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

[MR08] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In Bernstein et al. [BBD08], pages 147–191.

[MV10a] Daniele Micciancio and Panagiotis Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In Michael Mitzenmacher and Leonard J. Schulman, editors, *STOC*, pages 351–358. ACM, 2010.

[MV10b] Daniele Micciancio and Panagiotis Voulgaris. Faster exponential time algorithms for the shortest vector problem. In Moses Charikar, editor, *SODA*, pages 1468–1480. SIAM, 2010.

[NS09] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In Shai Halevi, editor, *CRYPTO*, volume 5677 of *Lecture Notes in Computer Science*, pages 18–35. Springer, 2009.

[Oka06] Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In Shai Halevi and Tal Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 80–99. Springer, 2006.

[OMA+99] Miyako Ohkubo, Fumiaki Miura, Masayuki Abe, Atsushi Fujioka, and Tatsuaki Okamoto. An improvement on a practical secret voting scheme. In Masahiro Mambo and Yuliang Zheng, editors, *ISW*, volume 1729 of *Lecture Notes in Computer Science*, pages 225–234. Springer, 1999.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *STOC*, pages 333–342. ACM, 2009.

[Pei10] Chris Peikert. An efficient and parallel gaussian sampler for lattices. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 2010.

[Poi98] David Pointcheval. Strengthened security for blind signatures. In Kaisa Nyberg, editor, *EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 391–405. Springer, 1998.

[PR06] Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In Shai Halevi and Tal Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 145–166. Springer, 2006.

[PR07] Chris Peikert and Alon Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In David S. Johnson and Uriel Feige, editors, *STOC*, pages 478–487. ACM, 2007.

[PS97] David Pointcheval and Jacques Stern. New blind signatures equivalent to factorization (extended abstract). In *ACM Conference on Computer and Communications Security*, pages 92–99, 1997.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer, 2008.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *STOC*, pages 187–196. ACM, 2008.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

[Reg10] Oded Regev. The learning with errors problem (invited survey). In *IEEE Conference on Computational Complexity*, pages 191–204. IEEE Computer Society, 2010.

[RHOAGZ07] Francisco Rodríguez-Henríquez, Daniel Ortiz-Arroyo, and Claudia García-Zamora. Yet another improvement over the mu-varadharajan e-voting protocol. *Comput. Stand. Interfaces*, 29(4):471–480, 2007.

[RS09] Markus Rückert and Dominique Schröder. Security of verifiably encrypted signatures and a construction without random oracles. In Hovav Shacham and Brent Waters, editors, *Pairing*, volume 5671 of *Lecture Notes in Computer Science*, pages 17–34. Springer, 2009.

[RS10a] Markus Rückert and Michael Schneider. Estimating the security of lattice-based cryptosystems. Cryptology ePrint Archive, Report 2010/137, 2010. `http://eprint.iacr.org/`.

[RS10b] Markus Rückert and Dominique Schröder. Fair partially blind signatures. In Daniel J. Bernstein and Tanja Lange, editors, *AFRICACRYPT*, volume 6055 of *Lecture Notes in Computer Science*, pages 34–51. Springer, 2010.

[RSS10] Markus Rückert, Michael Schneider, and Dominique Schröder. Generic constructions for verifiably encrypted signatures without random oracles or nizks. In Jianying Zhou and Moti Yung, editors, *ACNS*, volume 6123 of *Lecture Notes in Computer Science*, pages 69–86, 2010.

[Rüc09] Markus Rückert. Verifiably encrypted signatures from RSA without NIZKs. In Bimal K. Roy and Nicolas Sendrier, editors, *INDOCRYPT*, volume 5922 of *Lecture Notes in Computer Science*, pages 363–377. Springer, 2009.

[Rüc10a] Markus Rückert. Adaptively secure identity-based identification from lattices without random oracles. In Juan A. Garay and Roberto De Prisco, editors, *SCN*, volume 6280 of *Lecture Notes in Computer Science*, pages 345–362. Springer, 2010.

[Rüc10b] Markus Rückert. Lattice-based blind signatures. In Masayuki Abe, editor, *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 413–430. Springer, 2010.

[Rüc10c] Markus Rückert. Strongly unforgeable signatures and hierarchical identity-based signatures from lattices without random oracles. In Nicolas Sendrier, editor, *PQCrypto*, volume 6061 of *Lecture Notes in Computer Science*, pages 182–200. Springer, 2010.

[SE94] Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66:181–199, 1994.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.

[Sho] Victor Shoup. Number theory library (NTL) for C++. `http://www.shoup.net/ntl/`.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[SPC95] Markus Stadler, Jean-Marc Piveteau, and Jan Camenisch. Fair blind signatures. In Louis C. Guillou and Jean-Jacques Quisquater, editors,

EUROCRYPT, volume 921 of *Lecture Notes in Computer Science*, pages 209–219. Springer, 1995.

[SSTX09] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 617–635. Springer, 2009.

[ST01] Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 355–367. Springer, 2001.

[Wat09] John Watrous. Zero-knowledge against quantum attacks. *SIAM J. Comput.*, 39(1):25–58, 2009.

[WZ82] Wootters, W. K. and Zurek, W. H. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, October 1982.

[ZSNS03] Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. Efficient verifiably encrypted signature and partially blind signature from bilinear pairings. In Thomas Johansson and Subhamoy Maitra, editors, *INDOCRYPT*, volume 2904 of *Lecture Notes in Computer Science*, pages 191–204. Springer, 2003.