AUTOMATED ANALYSIS IN GENERIC GROUPS

Edvard Fagerholm

A DISSERTATION

in

Mathematics

Presented to the Faculties of

The University of Pennsylvania

in Partial Fulfillment of the Requirements for

the Degree of

Doctor of Philosophy

2015

Andre Scedrov, Professor of Mathematics

Supervisor of Dissertation

David Harbater, Christopher H. Browne Distinguished Professor

Professor of Mathematics

Graduate Group Chairperson

Dissertation Committee:

Andre Scedrov, Professor of Mathematics

Gilles Barthe, Professor

Ted Chinburg, Professor of Mathematics

Nadia Heninger, Professor of Computer Science

Dedicated to my wife Kaisa and our lovely daughter Ada

ACKNOWLEDGEMENTS

It feels somewhat strange that the period of my life called "doctoral studies" is coming to and end. In six years, surprisingly many things happen. I've had the chance to explore Philadelphia, which after some initial shock due to apartment choices, ended up being a wonderful place to live as well as a place that one could start calling home. Friends in the Ph.D. program have come and gone and are now dispersed around the globe. My daughter Ada was born and she has given me more joy than I ever could have imagined. Finally, through all this time, my wife Kaisa has always been there to support me even when it seemed like there was no light at the end of the tunnel.

Finishing a Ph.D., however, is something that one never could have done without help from many people. First and foremost, I must thank my advisor Andre Scedrov who has provided me with more flexibility than one could hope to ask for. My wife was able to accept a job offer with the UN in Rome, Italy, after our daughter was born, because Andre suggested that I visit the IMDEA Software Institute in Madrid. A big thank you also goes to my coauthors, especially Gilles Barthe at IMDEA who agreed to host me as well as Dario Fiore and Benedikt Schmidt who both have taught me a great deal and were always available for questions and discussions when needed.

Finally, I would never have survived without the support of other students in the program. In no particular order, I would especially like to thank Taisong Jing, Jonathan Kariv, Tyler Kelly and David Lonoff. Taisong was instrumental in introducing me to authentic Chinese food, John would never miss an opportunity to discuss anything related to powerlifting or to comment "do you mean kilos?" when you had made a record at the gym and told him what you lifted (which obviously was in pounds), Tyler is Tyler and, finally, David and I had many interesting discussions during the first two years of the Ph.D. program.

None of this would of course have been possible without funding. The work on this thesis was supported in part by ONR grant N00014-12-1-0914 with additional support from the AFOSR MURI "Science of Cyber Security: Modeling, Composition, and Measurement" and from NSF Grant CNS-0830949.

ABSTRACT

AUTOMATED ANALYSIS IN GENERIC GROUPS

Edvard Fagerholm

Andre Scedrov

This thesis studies automated methods for analyzing hardness assumptions in generic group models, following ideas of symbolic cryptography. We define a broad class of generic and symbolic group models for different settings—symmetric or asymmetric (leveled) k-linear groups—and prove "computational soundness" theorems for the symbolic models. Based on this result, we formulate a master theorem that relates the hardness of an assumption to solving problems in polynomial algebra. We systematically analyze these problems identifying different classes of assumptions and obtain decidability and undecidability results. Then, we develop automated procedures for verifying the conditions of our master theorems, and thus the validity of hardness assumptions in generic group models. The concrete outcome is an automated tool, the Generic Group Analyzer, which takes as input the statement of an assumption, and outputs either a proof of its generic hardness or shows an algebraic attack against the assumption.

Structure-preserving signatures are signature schemes defined over bilinear groups in which messages, public keys and signatures are group elements, and the verification algorithm consists of evaluating "pairing-product equations". Recent work on structure-preserving signatures studies optimality of these schemes in terms of the number of group elements needed in the verification key and the signature, and the number of pairing-product equations in the verification algorithm. While the size of keys and signatures is crucial for many applications, another aspect of performance is the time it takes to verify a signature. The most expensive operation during verification is the computation of pairings. However, the concrete number of pairings is not captured by the number of pairing-product equations considered in earlier work.

We consider the question of what is the minimal number of pairing computations needed to verify structure-preserving signatures. We build an automated tool to search for structure-preserving signatures matching a template. Through exhaustive search we conjecture lower bounds for the number of pairings required in the Type II setting and prove our conjecture to be true. Finally, our tool exhibits examples of structure-preserving signatures matching the lower bounds, which proves tightness of our bounds, as well as improves on previously known structure-preserving signature schemes.

CONTENTS

PREFACE

The motivation behind this thesis is to increase trust in security proofs in cryptography. In the last decade or so a development within cryptography has been to push the boundary on new features provided by cryptographic primitives [BF03, BBG05a, HL02, GPSW06, LOS$^+$10, Gen09], but at the same time relaxing the standard for security assumptions. While the security of cryptographic primitives used to rely on well-known and long-studied problems, such as the hardness of factoring and the discrete logarithm assumptions, cryptographers started inventing new assumptions while justifying the security of their assumptions in restricted computational models. The generic group model studied and extended in this thesis is one particularly common example while inventing new assumptions is especially prevalent within the field of pairing-based cryptography [KM10]. This has lead to a plethora of new security assumptions.

Worrying about this state-of-affairs is not new. In 2005, Halevi wrote a paper [Hal05], where he noted that "we generate more proofs than we carefully verify (and as a consequence some of our published proofs are incorrect)". Indeed, a paper was recently accepted to one of the flagship conferences in cryptography, ASIACRYPT, which had a faulty generic group proof [HS14, Fuc14, Pan14]. Halevi also noted that some of the reasons for this problem are social: "we mostly publish in conferences rather than journals". As a solution Halevi proposed writing a tool that could be used to verify the "mundane" parts of cryptographic proofs. Halevi's paper eventually led to Barthe et. al developing the CertiCrypt [BGZB09] and later the EasyCrypt [BGHZ11] proof assistants for writing mechanically verified cryptographic proofs. However, tools like EasyCrypt still have a fairly steep learning curve as well as requiring significant effort from the user compared to writing manual proofs. Unfortunately, this is currently a price to pay for mechanically verified proofs, which is shared by most proof assistants; formal proofs typically take an order-of-magnitude more time to write.

We note that attempts to formalize proofs is not just a development in e.g. the programming language and the cryptographic communities. A recent article by Avigad and Harrison [AH14] surveys the state of formally verified proofs in mathematics. A classic example of an unusually complicated proof is e.g. the Four Color Theorem proven by Appel and Haken [AH80] using a computer program to verify some 1,936 special cases as part of the proof. Traditional mathematical proofs of significant length and complexity include the proof of the Feit-Thompson theorem [FT62, FT63]. There has been an increased effort in recent years to use automated theorem provers and proof assistants to verify mathematical proofs. An example is the recently announced formal proof of the Feit-Thompson theorem after six years of effort by Gonthier et al. [GAA+13] that had followed a similar effort for the Four Color Theorem [Gon08]. Instead of verifying old theorems with very complicated proofs a different point-of-view has been taken by the *Univalent Foundations* project initiated by Voevodsky, Awodey and others [Uni13]. The aim of the project is to provide constructive foundations for contemporary abstract mathematics. This could be used to build a formalization of most of modern mathematics in a proof assistant such as Coq.

This thesis is an attempt to alleviate the problem of verifying new cryptographic assumptions. The concrete outcome is a tool, the Generic Group Analyzer, that takes as input a description of a cryptographic assumption or certain types of schemes and verifies the security or finds an attack. As an example, the tool can automatically find an attack against the signature scheme in the flawed ASIACRYPT paper mentioned above. It's also worth noting that the step to synthesis from verification is short. Having a tool that can automatically verify cryptographic schemes, we can exhaustively generate candidates from a pre-defined search space and use the verification tool to prune the search space of insecure schemes. One can then study the candidates that remain in order to find schemes satisfying the properties that were sought after. Finally, we note that such a tool is not just useful for practical discovery of new schemes, we can also use the tool to increase our belief in conjectures by looking for counterexamples.

This thesis is based on the following two papers:

- Gilles Barthe, Edvard Fagerholm, Dario Fiore, John C. Mitchell, Andre Scedrov and Benedikt Schmidt. Automated Analysis of Cryptographic Assumptions in Generic Group Models. In *CRYPTO 2014*, pages 354–368.

- Gilles Barthe, Edvard Fagerholm, Dario Fiore, Andre Scedrov, Benedikt Schmidt and Mehdi Tibouchi. Strongly-Optimal Structure Preserving Signatures from Type II Pairings: Synthesis and Lower Bounds. In *PKC 2015*.

Significant extensions have since been made to the CRYPTO 2014 paper with respect to supporting Laurent polynomials instead of just polynomials in the input language as well as extending the interactive solver to accept group elements as parameters to oracle queries. These extensions were both necessary prerequisites to be able to do the work in the PKC 2015 paper. Additionally, the tool has been extended to handle assumptions dealing with composite-order groups e.g. subgroup decision assumptions. We also develop the necessary theory that allow handling non-parametric assumptions with rational expressions as input, but this has not been implemented in the tool as this type of assumptions have not been widely used in the literature.

1 A PRIMER ON SOME TOPICS IN CRYPTOGRAPHY

We start by presenting the necessary background for the thesis. With respect to a mathematical background the reader is assumed to know the material covered in a standard undergraduate mathematics major. In particular, we assume knowledge of basic linear algebra, group theory and probability. From computer science, we assume that the reader is familiar with basics of algorithms, Turing machines and oracles at the level of Sipser [Sip06]. Additionally, we assume that the reader is familiar with some basic notions of cryptography, namely, security assumptions and reduction proofs at the level of Katz and Lindell [KL07]. All the other necessary background will either be covered in this chapter or given necessary references to when needed.

1.1 PAIRINGS AND MULTILINEAR MAPS

The Diffie-Hellman key exchange protocol between two parties A and B is one of the cornerstones of modern cryptography. To perform the key exchange, A and B agree on a group $\mathbb{G}$ of prime order p together with a generator g. They then proceed by choosing their respective secrets $a, b \in \mathbb{Z}/p\mathbb{Z}$ after which A sends g^a to B and B sends g^b to A. A then computes $h_A = (g^b)^a = g^{ab}$ and B computes $h_B = (g^a)^b = g^{ab}$, which is their shared secret. A passive eavesdropper is unable to compute the shared secret assuming the *Diffie-Hellman assumption* is hard, namely, given g, g^a, g^b computing g^{ab} should be infeasible in the group $\mathbb{G}$ with generator g.

The Diffie-Hellman key exchange protocol needs only one round, i.e. the parties send each other only one message, which is independent of any previously sent messages. A natural question to ask was then whether there exists a one-round key exchange protocol between three parties? It wasn't until 2000 that Joux [Jou00] settled this in the affirmative by a construction relying on pairings. Shortly thereafter, Boneh and Franklin [BF01] discovered the celebrated Boneh-Franklin Identity-Based Encryption scheme solving an old problem posed by Shamir [Sha85]. Since then, there has been a plethora of new cryptographic constructions based on pairings.

We now give the relevant definitions of pairings for this work and note that we only need to consider them as abstract operations for the rest of this work. However, we note that in general its important to understand the underlying instantiations and their limitations as recently pointed out by Menezes and Chatterjee [CM14].

Definition 1. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be cyclic groups of order n with generators g_1, g_2, g_T. A *bilinear group* is a tuple $(e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$, where $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a map satisfying

$$e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}.$$

We always assume that the pairing is *non-degenerate*, meaning that $e(g_1, g_2)$ is a generator of $\mathbb{G}_T$ and assume that $g_T = e(g_1, g_2)$.

Note that although the typical instantiation of a bilinear pairing comes from pairing operations on elliptic curves, where $\mathbb{G}_1$ and $\mathbb{G}_2$ is written using additive notation, it's customary in cryptography to use multiplicative notation for all groups.

Following Galbraith, Paterson and Smart [GPS08], we classify pairings into three types according to whether or not there exists efficiently computable isomorphisms $\mathbb{G}_1 \to \mathbb{G}_2$ or vice versa. We say a bilinear group is of *Type I* if there are efficiently computable isomorphisms $\mathbb{G}_1 \to \mathbb{G}_2$ and $\mathbb{G}_2 \to \mathbb{G}_1$, *Type II* if there is only an efficiently computable isomorphism $\mathbb{G}_2 \to \mathbb{G}_1$, and, *Type III* if there are no efficiently computable isomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$ in either direction. In the Type I setting we will in general write $\mathbb{G} = \mathbb{G}_1 = \mathbb{G}_2$, i.e. we may just assume that the source groups are equal. This simplifies notation. Furthermore, we will refer to Type I bilinear groups as *symmetric* while the Type II and III groups are called *asymmetric*. Finally, in order to use bilinear groups, we must be able to construct them. This leads to the following definition.

Definition 2. A *bilinear group generator* $\mathcal{G}$ is an efficient algorithm, which, on input of a security parameter 1^λ, returns a bilinear group description $(n, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, g_1, g_2)$, where $n = 2^{\Omega(\lambda)}$, ψ is a description of efficiently computable isomorphisms between the groups depending on the type and g_i is a generator of $\mathbb{G}_i$ for $i = 1, 2$.

For a bilinear group generator to be useful for cryptographic applications, it needs to generate group descriptions that satisfy some hardness assumption. In this thesis, we will always assume that the discrete logarithm problem is hard. Depending on the application, one might require other useful properties. For example, we might require the existence of efficient hash functions $H : \{0,1\}^* \to \mathbb{G}_2$, we might expect the group order to always be prime or always composite. Even though most papers on pairing-based cryptography only use the abstract description of a bilinear group, it's worth noting that for many combinations of additional requirements, we might not know of any efficient instantiations.

Definition 3. Let $\mathbb{G}_1, \ldots, \mathbb{G}_n, \mathbb{G}_T$ be cyclic groups of order n with generators $g_1, \ldots, g_n, g_T$. An *n-linear group* is a tuple $(e, \mathbb{G}_1, \ldots, \mathbb{G}_n, \mathbb{G}_T)$, where $e : \mathbb{G}_1 \times \cdots \times \mathbb{G}_n \to \mathbb{G}_T$ is a map satisfying

$$e(g_1^{a_1}, \ldots, g_n^{a_n}) = e(g_1, \ldots, g_n)^{a_1 \cdots a_n}.$$

Just as for bilinear groups, we assume that the pairing is *non-degenerate*, meaning that $e(g_1, \ldots, g_n)$ is a generator of $\mathbb{G}_T$, whenever each g_i generates $\mathbb{G}_i$. Again, we typically assume that $g_T = e(g_1, \ldots, g_n)$. When the value of n is not important, we will refer to n-linear maps as *multilinear*. As before, we will call an n-linear group *symmetric* if $\mathbb{G}_1 = \ldots = \mathbb{G}_n$ and otherwise it is called *asymmetric*.

Once the first important constructions based on bilinear maps had been discovered, Boneh and Silverberg [BS02] explored the question of what can be constructed if we could build useful n-linear maps? However, at the same time they were pessimistic that such maps could be constructed using algebraic geometry. It took 10 years before the first plausible cryptographically useful construction was discovered by Garg, Gentry and Halevi using ideal lattices [GGH13], which was quickly followed by a construction by Coron, Lepoint and Tibouchi [CLT13].

Unfortunately, the latter construction is now considered broken [CHL+14, CLT14]. Additionally, neither construction is useful in practical schemes when the degree of multilinearity is large. For example, the paper by Coron, Lepoint and Tibouchi presents a 7-multipartite Diffie-Hellman key exchange with 80-bits of security (security estimate prior to the above attack) using their construction. With this level of security, the public parameters that need to be shared between the parties, i.e. the chosen group elements etc., take some 2.5GB of space. Furthermore, the key exchange requires some 40 seconds of CPU time for each participant. Therefore, while the abstract view of multilinear maps is easy to work with, constructions can't be translated into practical use until the underlying instantiations have been significantly improved to the point where constructions as the one above takes a few milliseconds at most and would require an exchange of only a few hundred bytes of data.

The reader interested in how these instantiations of multilinear groups work is referred to the two papers mentioned above. However, an additional property worth noting is that both current instantiations actually instantiate something called a *leveled multilinear group*. A definition for the symmetric case follows.

Definition 4. Let $\mathbb{G}_1, \dots, \mathbb{G}_n$ be cyclic groups of order n with generators $g_1, \dots, g_n$. Assume that for each $i + j \leqslant n$, where $i, j \geqslant 1$ we have a map $e_{ij} : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_{i+j}$ satisfying

$$e_{ij}(g_i^a, g_j^b) = e_{ij}(g_i, g_j)^{ab}.$$

Then the tuple $(\mathbb{G}_1, \dots, \mathbb{G}_n, \{e_{ij}\})$ is called an n-*leveled multilinear group*.

An n-leveled multilinear group can be used to instantiate a symmetric n-linear group as follows. Set $\mathbb{G} = \mathbb{G}_1$, $\mathbb{G}_T = \mathbb{G}_n$ and define maps $e_k : \mathbb{G}_1^k \to \mathbb{G}_k$ by setting $e_2 = e_{1,1}$, and, recursively,

$$e_k(g_1^{a_1}, \dots, g_1^{a_k}) = e_{1,k-1}(g_1^{a_1}, e_{k-1}(g_1^{a_2}, \dots, g_1^{a_k})).$$

Then letting $e = e_n$, e is an n-linear map $e : \mathbb{G}^n \to \mathbb{G}_T$.

1.2 SIGNATURE SCHEMES

A problem that comes up in both digital contracts as well as software distribution is whether or not the content of a file is what it should be. For example when signing a digital contract, we may want to be certain that the file can't be tampered with while still claiming to represent the original agreement. When distributing software, we want to be sure that the file we are installing was actually distributed by the vendor and doesn't include malicious components added by a third party. The standard solution to this problem is a *signature scheme* defined below.

Definition 5. A *digital signature scheme* is a quadruple of efficient algorithms $(\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Sign}, \mathrm{Verify})$ defined as follows:

- Setup: Takes as input the security parameter, λ, and returns a *public parameter* $PP \leftarrow \mathrm{Setup}(\lambda)$.

- KeyGen: The key generation algorithm KeyGen takes PP as input and returns $(SK, VK) \leftarrow \mathrm{KeyGen}(PP)$, where SK is the secret *signing key* and VK the public *verification key*. We always assume that SK contains VK.

- Sign: The signing algorithm Sign takes as input PP, SK and a message M in the message space $\mathcal{M}$ defined by PP and returns a *signature* $\sigma \leftarrow \mathrm{Sign}(PP, SK, M)$.

- Verify: The verification algorithm Verify takes as input the public parameters PP, the verification key VK, a message M and the signature σ and returns a bit $b \leftarrow \mathrm{Verify}(PP, VK, M, \sigma)$. If $b = 1$ the algorithm accepts the signature and if $b = 0$ it rejects.

We require that the signature scheme is *correct*, i.e. for all correctly generated public parameters PP, key pairs (SK, VK), and messages M in the message space $\mathcal{M}$, we have $\mathrm{Verify}\left(PP, VK, M, \mathrm{Sign}(PP, SK, M)\right) = 1$.

Secure signature schemes are surprisingly difficult to construct in practice. Assume for example that $N = pq$ for some primes p, q and $ed \equiv 1 \pmod{\phi(N)}$, i.e.

(N, e, d) is the standard setup for RSA encryption. We can then set $pk = (N, e)$ and $sk = (N, d, e)$. One could then try to define a signature algorithm by setting

$$\sigma = \mathrm{Sign}(m, sk) = [m^d],$$

where $[m^d]$ denotes the value that m^d reduces to modulo N in the range $0, \ldots, N-1$. Verification could then be performed by checking whether

$$\mathrm{Verify}(m, \sigma, pk) = \sigma^d \overset{?}{\equiv} m.$$

Unfortunately, this would not be considered secure, since it's easy to see that if σ_1 is a valid signature for m_1 and σ_2 is a valid signature for m_2, then $\sigma_1\sigma_2$ is a valid signature for m_1m_2. In other words, the signature scheme is *malleable* by allowing us to construct valid signatures for messages not signed by the signer by knowing valid message signature pairs.

The standard method to guard against malleability is to take a hash function $H : \{0, 1\} \to \mathbb{Z}_N^*$ and hash messages in $\mathcal{M} = \{0, 1\}^*$ by setting

$$\sigma = \mathrm{Sign}(m, sk) = [H(m)^d].$$

A heuristic argument for the non-malleability of this signature scheme is that the hash function should destroy any algebraic relations. For example, a hash function should not satisfy relations like $H(m_1m_2) = H(m_2)H(m_2)$, which is the basis of the forgery mentioned above. The problem is that if we choose H to be a fixed hash function like Keccak, then we don't know how to prove security of the signature scheme. However, if we assume that H is a "random function", then we can reduce security to the RSA assumption. Such an argument that relies on the hash function being random is called a proof in the *Random Oracle Model* [BR93] and is a standard security heuristic for many cryptographic primitives.

There are two standard security definitions that we use for signatures in this work. The stronger definition is against an adaptive adversary that is allowed to adaptively query valid signatures on any messages of its choice from a signing oracle.

Definition 6 (EUF-CMA)**.** A signature scheme $(\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Sign}, \mathrm{Verify})$ is *existentially unforgeable under adaptive chosen message attack* (EUF-CMA) if for all non-uniform polynomial time algorithms $\mathcal{A}$, we have:

$$\Pr\left[\begin{array}{ll} PP \leftarrow \mathrm{Setup}(1^\lambda) & \\ (VK, SK) \leftarrow \mathrm{KeyGen}(PP) & : M \notin Q \ \wedge\ \mathrm{Verify}(PP, VK, M, \Sigma) = 1 \\ (M, \Sigma) \leftarrow \mathcal{A}^{\mathrm{Sign}(PP, SK, \cdot)}(PP, VK) & \end{array}\right] = \mathrm{negl}(\lambda),$$

where Q is the set of queries made by $\mathcal{A}$ to the signing oracle.

Sometimes it is also useful to prevent the adversary from issuing a new signature for a message that has already been signed. A signature scheme is *strongly existentially unforgeable* if it is hard to find a signature on a message that has not been signed before and also hard to find a new signature for a message that has already been signed. This notion, denoted by sEUF-CMA, is formally captured in the same way as the definition of EUF-CMA except for requiring $(M, \Sigma) \notin Q$ where Q is the set of message-signature pairs from $\mathcal{A}$'s queries to the signing oracle.

A much weaker definition only assumes an adversary that can get a set of signatures on random messages.

Definition 7 (EUF-RMA)**.** A signature scheme $(\mathrm{Setup}, \mathrm{KeyGen}, \mathrm{Sign}, \mathrm{Verify})$ is *existentially unforgeable under random message attack* (EUF-RMA) if for all non-uniform polynomial time algorithms $\mathcal{A}$, we have:

$$\Pr\left[\begin{array}{ll} PP \leftarrow \mathrm{Setup}(1^\lambda) & \\ (VK, SK) \leftarrow \mathrm{KeyGen}(PP) & : M \notin Q \ \wedge\ \mathrm{Verify}(PP, VK, M, \Sigma) = 1 \\ (M, \Sigma) \leftarrow \mathcal{A}^{\mathcal{O}(SK)}(PP, VK) & \end{array}\right] = \mathrm{negl}(\lambda),$$

where Q is the set of messages returned to $\mathcal{A}$ by the signing oracle $\mathcal{O}$.

Corresponding security notions for one-time (resp. n-time) signature schemes can be obtained by restricting the adversary to only calling the signing oracle once (resp. n times) in the above definitions.

2 THE GENERIC GROUP MODEL

Computing lower complexity bounds for any non-trivial algorithms is typically hard or impossible given currently available mathematical tools. The Millennium Prize problem **P** vs. **NP**, stated by Stephen Cook [Coo71], is a famous example of this type of problem and the lack of progress on it exemplifies this difficulty. The current state of affairs poses a major problem for cryptography, since it is well-known that proving that an encryption scheme is hard to break on the average implies $\mathbf{P} \neq \mathbf{NP}$. Therefore, unless we are trying to settle the **P** vs. **NP** problem, proving lower bounds in an unrestricted computational model is infeasible and we are forced to settle for the next best thing, i.e. define idealized models in which we can compute lower bounds. One can then hope that lower bounds in a reasonable simplified model translates to an actual lower bound, but no guarantee can of course be given. On the other hand, any scheme insecure in a simplified model is guaranteed to be insecure in the unrestricted model. In this chapter we will specifically explore idealized models for algorithms that operate on groups.

2.1 THE DEFINITIONS OF SHOUP AND MAURER

In purely algebraic terms, isomorphic groups are typically thought of as "equal". However, this does not take into account that the choice of representative from an isomorphism class of groups, i.e. the chosen encoding, typically has a major impact on the efficiency of algorithms that one can implement on it. A simple example is to take a prime q s.t. $p = (q-1)/2$ is prime and let $\mathbb{G}$ be the group of quadratic residues modulo q. We know that $\mathbb{G}$ is isomorphic to the additive group $\mathbb{Z}/p\mathbb{Z}$, since both have order p. However, if given generators $g \in \mathbb{G}$, $h \in \mathbb{Z}/p\mathbb{Z}$, and g^x as well as xh (h^x in multiplicative notation) for some randomly sampled

$x \in \mathbb{Z}/p\mathbb{Z}$, then computing x from g^x is thought to be hard [Bon98], while it's trivial to compute from xh by dividing by h. Thus, the discrete logarithm problem is hard in one instance while easy in another instance of a group belonging to the same isomorphism class. This difference is a consequence of the additional structure carried by $\mathbb{Z}/p\mathbb{Z}$, i.e. it's also a field, which is revealed when representing the elements in this form.

The idea of a *generic group* is a "generic representative" of an isomorphism class that hides any additional structure that a concrete representative might provide. In concrete terms this means that any algorithm that one can implement on a generic representative is implementable without modification on any representative of the same isomorphism class as long as the group operations are efficiently computable. Nechaev [Nec94] was the first to explore these ideas while alternative models were later proposed by Shoup [Sho97] and Maurer [Mau05]. The models of Shoup and Maurer are slightly more "generic", in the sense that certain algorithms that can be implemented in Nechaev's model can't be implemented in the other two models. Additionally, in cases where both Shoup's and Maurer's models apply, they have been proven to be equivalent by Jager and Schwenk [JS08].

We start by describing the model of Shoup. Let $(\mathbb{G}, +)$ be a cyclic group of order n and S be a set of bit strings of cardinality at least S. An *encoding function* is an injective map $\sigma : \mathbb{G} \to S$. A generic algorithm $\mathcal{A}$ operating on $\mathbb{G}$ is a probabilistic algorithm that takes as input a list of encodings $(\sigma(x_1), \dots, \sigma(x_k))$. During execution, $\mathcal{A}$ may compute $x_i \pm x_j$ by giving an oracle the indices i, j and a bit b. The oracle then computes $x_i + (-1)^b x_j$ and returns $\sigma(x_i + (-1)^b x_j)$ to $\mathcal{A}$, which appends $\sigma(x_i + (-1)^b x_j)$ to its current list of encodings. Finally, the output of $\mathcal{A}$ is denoted by $\mathcal{A}(x_1, \dots, x_k)$. We note that $\mathcal{A}$ is dependent on $\mathbb{G}$ and S, but can't depend on σ. In other words, it needs to be able to operate on any random encoding of the group elements, which prevents it from exploiting any special structure in the encodings. The runtime complexity of a generic algorithm $\mathcal{A}$ is the number of oracle queries made during execution.

In the model of Maurer, instead of having random bit strings represent the group elements, we refer to every given group element through a unique *handle*. Two distinct handles can refer to the same group element, so we provide an *equal-*

ity oracle that takes as parameters two handles and checks whether they internally represent the same element. Since any generic algorithm in Shoup's model can check for equality by himself, by just checking that the corresponding bit strings are the same, we account for this, by making the equality oracle calls free, i.e. we do not account for them in the runtime analysis. A more precise explanation follows. Let $(\mathbb{G}, +)$ be a cyclic group of order n. A generic algorithm $\mathcal{A}$ operates as follows. On initialization $\mathcal{A}$ is given integers $1, \dots, k$ as well as a group operation oracle that internally maintains a list L initialized to length k, with the ith element initialized to some $x_i \in \mathbb{G}$. When $\mathcal{A}$ wants to compute $x_i \pm x_j$, it sends the oracle the indices i and j as well as a bit b. The oracle computes $x_i + (-1)^b x_j$, appends the value to its internal list L and returns the index of the appended element. When $\mathcal{A}$ wants to check if two elements are equal, it queries the equality oracle with the indices corresponding to the elements. Finally, the output of $\mathcal{A}$ is denoted by $\mathcal{A}(x_1, \dots, x_k)$. The runtime complexity of a generic algorithm $\mathcal{A}$ is the number of oracle calls to the group operation oracle.

The following example illustrates both Shoup's and Maurer's model in action.

Example 8. Assume we are given the group $\mathbb{Z}/4\mathbb{Z}$ and and encoding function $\sigma : \mathbb{Z}/4\mathbb{Z}$, s.t.

$$\sigma(0) = 00 \quad \sigma(1) = 01, \quad \sigma(2) = 10, \sigma(3) = 11.$$

We illustrate the generic algorithm that gets as input $x_1 = 1$. It proceeds by computing recursively $x_n = x_{n-1} + x_1$ for $n = 2, 3, 4, 5$. In Shoup's model we send the group operation oracle the following four queries:

$$\mathcal{O}(01, 01, 0) = 10, \quad \mathcal{O}(10, 01, 0) = 11, \quad \mathcal{O}(11, 01, 0) = 00, \quad \mathcal{O}(00, 01, 0) = 01.$$

In Maurer's model the list L is initialized as $L = [1]$. To our queries we get back increasing handles and send the group operation oracle the four queries:

$$\mathcal{O}(1, 1, 0) = 2, \quad \mathcal{O}(2, 1, 0) = 3, \quad \mathcal{O}(3, 1, 0) = 4, \quad \mathcal{O}(4, 1, 0) = 5.$$

Note that at the end, the list maintained by the oracle will be

$$L = [1, 2, 3, 0, 1],$$

so if $\mathcal{A}$ would query the equality oracle with the indices 1 and 5 they would be reported back as equal.

From the previous example we see that in Shoup's model the algorithm $\mathcal{A}$ can try to "guess" encodings. For example $\mathcal{A}$ might have issued as it's first query $\mathcal{O}(00, 00, 0)$ receiving the response 00. Note that there would be no way to query the result from this computation in Maurer's model, since at this stage there is no valid handle that refers to an index in the list L containing the value 0. Therefore, the models are equivalent when S is large enough with respect to the group $\mathbb{G}$ that the probability of $\mathcal{A}$ guessing a bit string that refers to a valid element of $\mathbb{G}$ is negligible. Note that even though $\mathcal{A}$ may internally perform any amount of computation, it still can't guess bit strings if they are sparse, since each guess would require a call to the oracle, which would be accounted for in its runtime complexity.

In what follows we will discuss the extended generic group model of [BFF$^+$14], which in the special case of a single group reduces to the model by Maurer.

2.2 GROUP SETTINGS

The generic group construction by Nechaev, Shoup and Maurer were restricted to single groups, but the constructions have since been generalized to bilinear groups [BBG05b, Boy08, Fre10]. To accommodate for (leveled) multilinear maps, as well as possibly more complicated scenarios, we introduce the notion of a group setting.

Definition 9. A *group setting* is a tuple $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$, where I is an index set, $\mathcal{G} = \{\mathbb{G}_i\}_{i \in I}$ a family of cyclic groups of order n indexed by I, Φ is a set of isomorphisms $\varphi : \mathbb{G}_i \to \mathbb{G}_j$, and $\mathcal{E}$ a set of admissible k-linear maps $e : \mathbb{G}_{i_1} \times \cdots \times \mathbb{G}_{i_k} \to \mathbb{G}_{i_{k+1}}$. We note that the k is not fixed, so the maps in $\mathcal{E}$ do not need to have the same multilinearity. We often drop the index set I when it is obvious.

A group setting simply describes a collection of cyclic groups of equal orders together with the efficiently computable maps between them. These efficiently computable maps can either be multilinear maps or isomorphisms between individual groups. The following examples illustrate different settings.

Example 10. A prime order p bilinear group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ in the Type II setting is represented by $(p, \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T\}, \{1, 2, T\}, \{\mathbb{G}_2 \to \mathbb{G}_1\}, \{e\})$.

Example 11. A pair of cyclic groups $(\mathbb{G}_1, \mathbb{G}_2)$ of order n with an isomorphism from $\mathbb{G}_1 \to \mathbb{G}_2$ is represented by $(n, \{\mathbb{G}_1, \mathbb{G}_2\}, \{1, 2\}, \{\mathbb{G}_1 \to \mathbb{G}_2\}, \emptyset)$.

From the latter example we see that a group setting doesn't necessary have to represent a collection of groups used in pairing-based cryptography. Finally, since group settings will be used as internal hidden group representations in our generalized generic group construction, our goal will be to choose "nice" internal representations and for this we introduce the concept of an isomorphism of group settings.

Definition 12. Two group settings $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$ and $\mathcal{GS}' = (n, \mathcal{G}', I, \Phi', \mathcal{E}')$, with $\mathcal{G} = \{\mathbb{G}_i\}_{i \in I}$, $\mathcal{G}' = \{\mathbb{G}'_i\}_{i \in I}$ are *isomorphic* if there exists isomorphisms $\eta_i : \mathbb{G}_i \to \mathbb{G}'_i$ and bijections $\mu : \Phi \to \Phi'$ and $\nu : \mathcal{E} \to \mathcal{E}'$ such that for all $e \in \mathcal{E}$ and $\varphi \in \Phi$ the following diagrams are well-defined and commute

$$\begin{tikzcd}
\prod_{j=1}^{k} \mathbb{G}_{i_j} \arrow[r, "e"] \arrow[d, "\prod_{j=1}^{k} \eta_{i_j}"'] & \mathbb{G}_{i_{k+1}} \arrow[d, "\eta_{i_{k+1}}"] \\
\prod_{j=1}^{k} \mathbb{G}'_{i_j} \arrow[r, "\nu(e)"'] & \mathbb{G}'_{i_{k+1}}
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathbb{G}_i \arrow[r, "\varphi"] \arrow[d, "\eta_i"'] & \mathbb{G}_j \arrow[d, "\eta_j"] \\
\mathbb{G}'_i \arrow[r, "\mu(\varphi)"'] & \mathbb{G}'_j
\end{tikzcd} .$$

For the isomorphism class of cyclic groups of order n, the most elementary representative is arguably the group $(\mathbb{Z}/n\mathbb{Z}, +)$. As one would expect, for any group setting consisting of groups of order n, one may construct an isomorphic group setting, where each group is $(\mathbb{Z}/n\mathbb{Z}, +)$. The following result makes this precise.

Theorem 13. *Let $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$ be a group setting and $\mathcal{G}' = \{\mathbb{G}'_i \mid i \in I\}$ a family of groups such that $\mathbb{G}_i \cong \mathbb{G}'_i$. If $\eta_i : \mathbb{G}_i \to \mathbb{G}'_i$ is any set of isomorphisms, then there is a group setting $\mathcal{GS} = (n, \mathcal{G}', I, \Phi', \mathcal{E}')$ such that the collection $\{\eta_i : \mathbb{G}_i \to \mathbb{G}'_i\}$ of group isomorphisms can be extended to an isomorphism of group settings $\mathcal{GS} \to \mathcal{GS}'$.*

Proof. Assume we are given isomorphisms $\eta_i : \mathbb{G}_i \to \mathbb{G}'_i$ for all $i \in I$. Define $\Phi' = \{\eta_j \circ \varphi \circ \eta_i^{-1} \mid \varphi : \mathbb{G}_i \to \mathbb{G}_j \in \Phi\}$ and $\mathcal{E}' = \{\eta_{i_{k+1}} \circ e \circ \prod_{j=1}^{k} \eta_{i_j}^{-1} \mid e : \mathbb{G}_{i_1} \times \cdots \times \mathbb{G}_{i_k} \to \mathbb{G}_{i_{k+1}} \in \mathcal{E}\}$. With these definitions, the correct diagrams trivially commute. □

The following corollary gives us a nice concrete normal form for all the group settings we will encounter. An important point is that we may fix the generators of each group to be 1. When dealing with composite order groups, we will typically combine this normal form with an application of the previous theorem and the canonical isomorphism $\mathbb{Z}/n\mathbb{Z} \cong \prod_{i=1}^{k} \mathbb{Z}/p^{r_i}\mathbb{Z}$, where $n = p_1^{r_1} \cdots p_k^{r_k}$, since the product group form is more convenient in this case.

Corollary 14. *Let $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$ be a group setting where each $\mathbb{G}_i \in \mathcal{G}$ is cyclic of degree n. Then $\mathcal{GS}$ is isomorphic to a group setting $\widetilde{\mathcal{GS}} = (n, \tilde{\mathcal{G}}, I, \tilde{\Phi}, \tilde{\mathcal{E}})$, where each $\tilde{\mathbb{G}}_i \in \tilde{\mathcal{G}}$ is equal to $\mathbb{Z}/n\mathbb{Z}$. Furthermore, if each $\mathbb{G}_i \in \mathcal{G}$ has a fixed generator $g_i \in \mathbb{G}_i$, we may assume that the isomorphism maps g_i to $1 \in \mathbb{Z}/n\mathbb{Z}$.*

Proof. Let each $\eta_i : \mathbb{G}_i \to \mathbb{Z}/n\mathbb{Z}$ in the previous theorem be the isomorphism taking the fixed generator of $\mathbb{G}_i$ to 1. □

We note that in most scenarios arising in practice, we assume that the generators are chosen in a way such that if $e : \mathbb{G}_{i_1} \times \cdots \times \mathbb{G}_{i_k} \to \mathbb{G}_{i_{k+1}} \in \mathcal{E}$ is a k-linear map and each $\mathbb{G}_{i_j}$ has a fixed generator g_{i_j}, then the equation

$$e(g_{i_1}, \ldots, g_{i_k}) = g_{i_{k+1}}$$

holds and in addition for all isomorphism $\mathbb{G}_i \to \mathbb{G}_j$, we have $g_i \mapsto g_j$. Of course, this might not always be possible, if different maps in $\mathcal{E}$ and Φ pose conflicting requirements. However, when this is possible, then the isomorphic group setting provided by the corollary gives a particularly simple group setting, where each pairing $\tilde{e} \in \tilde{\mathcal{E}}$ and isomorphism $\tilde{\varphi} \in \tilde{\Phi}$ will satisfy

$$\tilde{e}(1, \ldots, 1) = 1, \quad \tilde{\varphi}(1) = 1.$$

2.3 extending maurer's definition

We extend the generic group model of Maurer by defining it as a computational model, where a probabilistic algorithm $\mathcal{A}$ is given access to an oracle $\mathcal{O}$, which performs computations on a group setting. In order to give a precise definition, we start by explaining how the oracle operates.

Let $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$ be a group setting. The oracle $\mathcal{O}$ internally maintains for each $i \in I$ a set of lists L_i, where each L_i contains group elements of the group $\mathbb{G}_i$. $\mathcal{O}$ supports the following operations:

1. **Group operations on the groups** $\mathbb{G}_i$: The oracle takes as input a group index i, two indices j, l in the list L_i, as well as a bit b, and computes $L_i[j] + (-1)^b L_i[l]$, appends the value to L_i and returns the index in L_i of the newly appended element.

2. **Isomorphisms of** Φ: The oracle takes as input a description of an isomorphism $\varphi : \mathbb{G}_i \to \mathbb{G}_j \in \Phi$ as well as an index l in L_i, appends $\varphi(L_i[l])$ to L_j and returns the index in L_j of the newly appended element.

3. k**-linear maps of** $\mathcal{E}$: The oracle takes as input a description of a map $e : \mathbb{G}_{i_1} \times \cdots \times \mathbb{G}_{i_k} \to \mathbb{G}_{i_{k+1}} \in \mathcal{E}$, indices $j_1, \ldots, j_k$ in $L_{i_1}, \ldots, L_{i_k}$, appends $e(L_{i_1}[j_1], \ldots, L_{i_k}[j_k])$ to $L_{i_{k+1}}$ and returns the index in $L_{i_{k+1}}$ of the newly appended element.

4. **Equality queries**: The oracle takes as input a group index i as well as two indices j, k in L_i and returns true if $L_i[j] = L_i[k]$ and false if $L_i[j] \neq L_i[l]$.

We omit the rigorous details of how to specify an input language for $\mathcal{O}$ that let's it distinguish the requested operation as well as parse its input. If the oracle receives a malformed input, by e.g. receiving an index that is out-of-bounds for the current state of one of its internal lists, the oracle returns some special symbol $\perp$ indicating an error.

Definition 15. The *extended generic group model* is an idealized computational model, where an algorithm $\mathcal{A}$ is given access to an oracle $\mathcal{O}$ corresponding to a group setting $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$ as described above. The lists L_i, $i \in I$, are each initialized to contain n_i elements of $\mathbb{G}_i$ according to some distributions $\mathcal{D}_i$. The group setting $\mathcal{GS}$ and the values n_i are assumed to be known by $\mathcal{A}$ together with descriptions of each $\mathcal{D}_i$. The initial values of the lists L_i is the *input* of $\mathcal{A}$. The *runtime* of the algorithm $\mathcal{A}$ is the number of queries made to the oracle $\mathcal{O}$, not counting equality queries. The oracle $\mathcal{O}$ is called a *generic group*.

Note that a generic group is a probability distribution in the sense that its behavior will be determined by the initial state of the lists L_i provided by the joint distribution of each $\mathcal{D}_i$. Particular initial values of the lists L_i defines an *instance* of a generic group. Another important detail is that we only count the number of oracle queries made by $\mathcal{A}$. $\mathcal{A}$ could internally perform any amount of computation without doing queries to $\mathcal{O}$ and this will not impact our runtime analysis.

As discussed in section 2.1, we want a generic group to somehow represent its isomorphism class in the sense that the particular choice of representative doesn't matter. The following theorem captures this property.

Theorem 16. *Let $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$ and $\mathcal{GS}' = (n, \mathcal{G}', I, \Phi', \mathcal{E}')$ be isomorphic group settings given by the isomorphisms $\eta_i : \mathbb{G}_i \to \mathbb{G}_i'$ and $\mathcal{O}$, $\mathcal{O}'$ the corresponding generic groups as defined above. Let $\mathcal{D}_i$ be a distribution sampling the initial value of the state L_i of $\mathcal{O}$. We define $\mathcal{D}_i'$ as the distribution, where we apply η_i to each element of the sampled list L_i and assume that the initial state of $\mathcal{O}'$ is sampled by $\mathcal{D}_i'$. Then given any algorithm $\mathcal{A}$, it cannot distinguish whether it's given a random instance of the oracle $\mathcal{O}$ or $\mathcal{O}'$. Furthermore, applying the same transformation to an already sampled instance of a generic group gives an indistinguishable instance of a generic group.*

Proof. Let $\{L_i'\}$ be the image of $\{L_i\}$, when the isomorphism η_i is applied to each element of L_i for each i. By the definition of an isomorphism of group settings, the generic group instance of $\mathcal{O}$ initialized with $\{L_i\}$ and $\mathcal{O}'$ initialized with $\{L_i'\}$ will respond identically to any sequence of queries. □

What the previous theorem says is that we can use isomorphisms of group settings to change the internal representation of the groups as long as the initial states are sampled by distributions compatible as in the theorem. This is particularly important for automated analysis, since we may after an appropriate translation assume that a generic group $\mathcal{O}$ corresponding to the group setting $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$ always satisfies $\mathbb{G}_i = \mathbb{Z}/n\mathbb{Z}$ for all $i \in I$. Therefore, representing the groups on a computer is trivial.

2.4 THE SCHWARZ-ZIPPEL LEMMA

In this section we prove for completeness the Schwartz-Zippel lemma, which is a key ingredient in the standard proof technique used for proving hardness results in the generic group model. In order to analyze composite-order groups, we prove the following slightly more general version.

Lemma 17 (Schwartz-Zippel Lemma)**.** *Let* $P \in (\mathbb{Z}/n\mathbb{Z})[X_1, \ldots, X_k]$ *be a polynomial of total degree* d*, where* $n = p_1 \cdots p_r$ *is a product of distinct primes. Then the probability that an independently chosen uniformly random element* $(a_1, \ldots, a_k) \in (\mathbb{Z}/n\mathbb{Z})^k$ *is a root of* P *is bounded above by* $\frac{d^r}{n}$*.*

Proof. We start by first proving the lemma when $n = p$ is prime by doing induction on k. If $k = 1$, then P can have at most d roots, so a uniformly chosen element in $\mathbb{Z}/p\mathbb{Z}$ is a root with probability d/p. For $k > 1$, we may write the polynomial in the form

$$P(X_1, \ldots, X_k) = \sum_{i=0}^{d} P_i(X_1, \ldots, X_{k-1}) X_k^i.$$

Let i be the highest i, s.t. $P_i \neq 0$ and denote by $U_1, \ldots, U_k$ independent uniform distributions on $\mathbb{Z}/p\mathbb{Z}$, by A the event $P(U_1, \ldots, U_k) = 0$ and by B the event $P_i(U_1, \ldots, U_{k-1}) = 0$. By definition, $P(a_1, \ldots, a_{k-1}, X_k)$ is of degree at most i and $\deg P_i \leqslant d - i$, so by induction

$$\Pr[P(U_1, \ldots, U_k) = 0] = \Pr[A \mid B] \Pr[B] + \Pr[A \mid \neg B] \Pr[\neg B] \leqslant \frac{d-i}{p} + \frac{i}{p} = \frac{d}{p}.$$

The general case follows, since by the Chinese Remainder Theorem the values of some $a \in \mathbb{Z}/n\mathbb{Z}$ modulo distinct prime factors p, q of n are independent. Therefore, random elements $a_1, \ldots, a_k \in \mathbb{Z}/n\mathbb{Z}$ are a root of $P \in (\mathbb{Z}/n\mathbb{Z})[X_1, \ldots, X_n]$ if they are independently roots modulo each prime factor. It follows that if

$n = p_1 \cdots p_r$ and $U_1, \dots, U_k$ are independent uniformly distributed random variables on $\mathbb{Z}/n\mathbb{Z}$, then

$$\Pr[P(U_1, \dots, U_k) = 0] \leqslant \frac{d^r}{p_1 \cdots p_r} = \frac{d^r}{n}.$$

□

Corollary 18. *Let* $P \in (\mathbb{Z}/n\mathbb{Z})[X_1, \dots, X_k]$ *be a polynomial of total degree at most* d*, where* $n = p_1 \cdots p_r$ *is a product of distinct primes. Then* P *has at most* $d^r n^{k-1}$ *roots.*

Proof. If there existed a polynomial with more roots, then a randomly sampled element in $(\mathbb{Z}/n\mathbb{Z})[X_1, \dots, X_k]$ would be a root of P with probability higher than d^r/n. □

It turns out that most hardness proofs in the generic group model in the literature omit rather sloppily the fact that any algorithm with access to a generic group instance performs queries in an adaptive fashion, i.e. the choice of the next query depends on the result of the previous query. It turns out, however, that omitting adaptiveness doesn't change the typical probability bounds, so this doesn't lead to incorrect results. However, to fix this lack of rigor, we derive the following result from the Schwartz-Zippel lemma and use it as the main ingredient of our proofs.

Lemma 19. *Let* $n = p_1 \cdots p_r$ *be a product of distinct primes. Assume we have a game, where we sample* $x \in (\mathbb{Z}/n\mathbb{Z})^k$ *and assume that an adversary is allowed to adaptively choose* q *polynomials of* $(\mathbb{Z}/n\mathbb{Z})[X_1, \dots, X_k]$ *of degree at most* d *without knowing* x*. After each choice we tell whether or not* x *is a root of the chosen polynomial and the adversary wins if it finds a polynomial, such that* x *is a root. Then, in the previously described game, the winning probability of the adversary is bounded above by* qd^r/n*.*

Proof. From the previous corollary, we know that any chosen polynomial $P \in (\mathbb{Z}/n\mathbb{Z})[X_1, \dots, X_k]$ of degree d has at most $d^r n^{k-1}$ roots. Therefore, we can change the game, so that the adversary may adaptively choose sets of size $d^r n^{k-1}$ and it wins if x belongs to one of these sets. Clearly, any upper bound in the winning probability in this game upper bounds the winning probability in the original game.

If the adversary is allowed one query, then clearly the winning probability is bounded above by

$$\frac{d^r n^{k-1}}{n^k} = \frac{d^r}{n}.$$

If we allow q choices, then we either win during the first $q-1$ or we do not win on the first $q-1$ choices and win on the qth choice when there are at least $n^k - (q-1)d^r n^{n-1}$ possible values left for x. By induction the winning probability is then

$$\frac{(q-1)d^r}{n} + \frac{d^r n^{k-1}}{n^k - (q-1)d^r n^{k-1}}\left(1 - \frac{(q-1)d^r}{n}\right) = \frac{qd^r}{n}.$$

□

As mentioned above, note that the upper-bound is the same as the upper bound that we get from Schwartz-Zippel by non-adaptively choosing q polynomials of degree d.

2.5 A FEW PROOF EXAMPLES

In this section we prove two well-known hardness results in the classical generic group model. Both problems only concern single groups, so the extended generic group model reduces to the generic group model of Maurer, but we will present them in the terminology of the previous section to better illustrate the concept to the reader. Namely, we start with the discrete logarithm problem, a computational assumption, and finish off with a decisional assumption, the Decisional Diffie-Hellman problem. The proofs highlight the main proof methods in the generic group model, which are generalized in the next section to the extended generic group model. Furthermore, the examples serve to highlight that the proofs follow a very specific template, which makes full automation possible.

As a warmup to the proof, we discuss a key step required for understanding the proof method. Assume that we have the group setting $\mathcal{GS} = (n, \{\mathbb{Z}/n\mathbb{Z}\}, \{1\}, \emptyset, \emptyset)$ and $\mathcal{O}$ is a corresponding generic group instance, where we initialize L_1 to, say, $[1, x, y, z]$, where $x, y, z \leftarrow \mathbb{Z}/n\mathbb{Z}$. However, we can also obtain an oracle $\mathcal{O}'$ by setting $L_1 = [1, X, Y, Z]$, where X, Y, Z are formal variables and we additionally sample $x, y, z \leftarrow \mathbb{Z}/n\mathbb{Z}$. $\mathcal{O}'$ then operates as follows:

- On a query asking to compute $L_1[i] + (-1)^b L_1[j]$, we compute $L_1[i] + (-1)^b L_1[j] \in (\mathbb{Z}/n\mathbb{Z})[X, Y, Z]$, i.e. we add/subtract the corresponding formal polynomials and append the resulting formal polynomial to L_1.

- On an equality query asking to compare $L_1[i]$ to $L_1[j]$, we plug in x, y, z for X, Y, Z and compare the corresponding values, i.e. we check whether $L_1[i](x, y, z) = L_1[j](x, y, z)$.

Note that it makes no difference for the equality check whether we plug in for X, Y, Z immediately on initialization or only when performing equality checks. Therefore, the two oracles behave exactly the same way, in other words, they are indistinguishable.

Example 20 (The Discrete Logarithm Problem)**.** Let p be a prime and $\mathbb{G}$ a group of order p. The discrete logarithm problem is defined as the problem where an adversary is given (g, g^x), where $g \leftarrow \mathbb{G}$ and $x \leftarrow \mathbb{Z}/p\mathbb{Z}$, and is asked to compute x. We formulate the problem in the generic group model below and prove a generic lower bound for the success probability of any adversary.

Let p be a prime and assume that we have the group setting $\mathcal{GS} = (p, \{\mathbb{G}_1\}, \{1\}, \emptyset, \emptyset)$, where the group operation in $\mathbb{G}_1$ is written additively. We define $\mathcal{D}_1$ as the distribution that sets $L_1 = [x, yx]$, where $x \leftarrow \mathbb{G}_1$ and $y \leftarrow \mathbb{Z}/p\mathbb{Z}$. The *discrete logarithm problem* is then the problem of computing y given a random instance of the generic group $\mathcal{O}$ defined by $\mathcal{GS}$ and initialized by $\mathcal{D}_1$. Note that by Theorem 16 we may assume that $\mathcal{GS} = (p, \{1\}, \{\mathbb{Z}/p\mathbb{Z}\}, \emptyset, \emptyset)$. In which case $\mathcal{D}_1$ becomes the distribution, where both x, y are sampled independently from $\mathbb{Z}/p\mathbb{Z}$.

The adversary wins if it correctly returns the value of y. If we denote by $\mathcal{O}(x, y)$ the oracle, where x and y has been given fixed values, and by X, Y independent uniform distributions on $\mathbb{Z}/p\mathbb{Z}$, then the winning probability of $\mathcal{A}$ is

$$\Pr[\mathcal{A}^{\mathcal{O}(X,Y)} = Y]$$

where the probability is taken over the random variables X and Y as well as any random coins used by $\mathcal{A}$. The winning probability can be upper-bounded based on the number of group operation queries made by $\mathcal{A}$. We are interested in the number of queries that $\mathcal{A}$ must perform in order to have

$$\Pr[\mathcal{A}^{\mathcal{O}(X,Y)} = Y] \geqslant c > 0$$

for some fixed constant c.

The proof now follows as a sequence of games, where each game is played by an adversary $\mathcal{A}$ querying an oracle $\mathcal{O}$ some q times and finally returning an element $y \in \mathbb{Z}/p\mathbb{Z}$. In the sequence of games, we make small changes to the oracle $\mathcal{O}$ analyzing the winning probabilities of $\mathcal{A}$ in adjacent games in the sequence. At step i we denote the oracle by $\mathcal{O}_i$. In the final game, $\mathcal{A}$ obtains no information about the sampled value Y, so it will have to resort to pure guessing, while in the first game, the oracle is the actual generic group.

Game 1: Here $\mathcal{O}_1$ is an instance of the generic group defined by $\mathcal{GS}$ with $\mathcal{D}_1$ used to initialize L_1. In this case

$$\Pr[\mathcal{A}^{\mathcal{O}_1(X,Y)} = Y] = \varepsilon.$$

Game 2: We change Game 1, so that X is sampled in $(\mathbb{Z}/p\mathbb{Z}) \setminus \{0\}$ instead. In this case

$$\Pr[\mathcal{A}^{\mathcal{O}_2(X,Y)} = Y] \approx \varepsilon,$$

since there is a negligible probability that X is sampled to be 0.

Game 3: We first sample the value of $x \leftarrow (\mathbb{Z}/p\mathbb{Z}) \setminus \{0\}$ and $y \leftarrow \mathbb{Z}/p\mathbb{Z}$, but then apply the group setting isomorphism induced by $x \mapsto 1$ on the oracle. This is equivalent to changing $\mathcal{D}_1$ to sample $L_1 = [1, y]$, ignoring x. By theorem 16 the resulting oracle $\mathcal{O}_3$, will behave exactly as in Game 2. Therefore,

$$\Pr[\mathcal{A}^{\mathcal{O}_2(X,Y)} = Y] = \Pr[\mathcal{A}^{\mathcal{O}_3(X,Y)} = Y].$$

Game 4: We instead set $L_1 = [1, Z]$, where Z is a formal variable and sample $y \leftarrow \mathbb{Z}/p\mathbb{Z}$. When $\mathcal{O}$ performs group operations on elements of L_1 it does them in the ring $(\mathbb{Z}/p\mathbb{Z})[Z]$. When an equality query is made for the handles i and j, we substitute y for Z, i.e. check if $L_1[i](y) = L_1[j](y)$. The difference here to Game 3 is that instead of plugging in y for Z during initialization, we do it prior to equality checks. As discussed above, this makes no difference, so that

$$\Pr[\mathcal{A}^{\mathcal{O}_3(X,Y)} = Y] = \Pr[\mathcal{A}^{\mathcal{O}_4(X,Y)} = Y].$$

Game 5: We change Game 4, so that when an equality query is made for handles i and j, we instead check if $L_1[i] = L_1[j]$. In other words, we do not plug in y for Z. Note that the value of y is not used at all by $\mathcal{O}_5$, so $\mathcal{A}$ can only blindly guess. It follows that

$$\Pr[\mathcal{A}^{\mathcal{O}_5(X,Y)} = Y] = \frac{1}{p}$$

and the interesting step in the proof is bounding the difference

$$\left|\Pr[\mathcal{A}^{\mathcal{O}_5(X,Y)} = Y] - \Pr[\mathcal{A}^{\mathcal{O}_4(X,Y)} = Y]\right|$$

which is the main step in the proof.

We note that after q group operation queries, L_1 will contain $q + 2$ elements. Let E denote the event that at some point during the execution of $\mathcal{A}$, there will be two handles i, j, such that

$$L_1[i] - L_1[j] \neq 0, \quad (L_1[i] - L_1[j])(y) = 0.$$

If event E never happens, then $\mathcal{O}_4$ and $\mathcal{O}_5$ will answer identically to all queries. If we denote by A the event $\mathcal{A}^{\mathcal{O}_5(X,Y)} = Y$ and by B the event $\mathcal{A}^{\mathcal{O}_4(X,Y)} = Y$, we may write

$$\begin{aligned}|\Pr[A] - \Pr[B]| &= |\Pr[A \cap E] + \Pr[A \cap E^c] - (\Pr[B \cap E] + \Pr[B \cap E^c])| \\ &= |(\Pr[A \mid E] - \Pr[B \mid E]) \Pr[E] + (\Pr[A \mid E^c] - \Pr[B \mid E^c]) \Pr[E^c]| \\ &= |(\Pr[A \mid E] - \Pr[B \mid E]) \Pr[E]| \\ &\leqslant \Pr[E].\end{aligned}$$

However, if $\mathcal{A}$ makes q group operation queries, then $\Pr[E]$ is bounded above by the probability that an adversary adaptively choosing $(q+2)^2/2$ polynomials of degree 1 finds a polynomial for which y is a root. This probability we know an upper bound for from Lemma 19, so that

$$\left|\Pr[\mathcal{A}^{\mathcal{O}_5(X,Y)} = Y] - \Pr[\mathcal{A}^{\mathcal{O}_4(X,Y)} = Y]\right| \leqslant \Pr[E] \leqslant \frac{(q+2)^2}{2p}.$$

Summarizing, we have that

$$\Pr[\mathcal{A}^{\mathcal{O}_1(X,Y)} = Y] \approx \Pr[\mathcal{A}^{\mathcal{O}_4(X,Y)} = Y], \ \ \Pr[\mathcal{A}^{\mathcal{O}_5(X,Y)} = Y] = \frac{1}{p}$$

and

$$\left|\Pr[\mathcal{A}^{\mathcal{O}_5(X,Y)} = Y] - \Pr[\mathcal{A}^{\mathcal{O}_4(X,Y)} = Y]\right| \leqslant \frac{(q+2)^2}{2p},$$

which implies that

$$\Pr[\mathcal{A}^{\mathcal{O}_1(X,Y)} = Y] \leqslant \frac{1}{p} + \frac{(q+2)^2}{2p}.$$

Therefore, if we want the success probability to be larger than a positive constant $c > 0$, we must have

$$c \leqslant \frac{1}{p} + \frac{(q+2)^2}{2p}.$$

It follows that we need at least $\Omega(\sqrt{p})$ queries.

In the next example we show how we can handle decisional assumptions by using the exact same proof technique.

Example 21 (The Decisional Diffie-Hellman Problem)**.** Let p be a prime and $\mathbb{G}$ a group of order p. The Decisional Diffie-Hellman problem is defined as the problem where an adversary is given the tuple (g, g^x, g^y, g^z), where either $z = xy$ or z is $z \leftarrow \mathbb{Z}/p\mathbb{Z}$ is uniformly random.

We can directly skip the two first steps in the previous proof by the exact same argument, so assume that we have the group setting $\mathcal{GS} = (p, \{\mathbb{Z}/p\mathbb{Z}\}, \{1\}, \emptyset, \emptyset)$. Denote by $\mathcal{O}_1^L$ the oracle, where $L_1 = [1, x, y, xy]$ and by $\mathcal{O}^R$ the oracle, where $L_1 = [1, x, y, z]$, where $x, y, z \leftarrow \mathbb{Z}/p\mathbb{Z}$. We assume that an adversary wins if given an instance of $\mathcal{O}_1^L$ it returns 0 or if it returns 1 when given an instance of $\mathcal{O}_1^R$. Again, denote by X, Y, Z independent uniformly distributed random variables on $\mathbb{Z}/p\mathbb{Z}$. Therefore, if an adversary $\mathcal{A}$ does at most q queries, using the notation from the previous example, we want to bound the difference

$$\left| \Pr[\mathcal{A}^{\mathcal{O}_1^L(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_1^R(X,Y,Z)} = 0] \right| .$$

We first look at the following sequence of games:

Game 1: $\mathcal{A}$ is given an instance of the oracle $\mathcal{O}_1^L$.

Game 2: $\mathcal{A}$ is given an instance of the oracle $\mathcal{O}_2^L$, where we instead treat x, y, z as formal variables and plug in for them during equality check queries.

Game 3: $\mathcal{A}$ is given an instance of $\mathcal{O}_3^L$, where the oracle operates as $\mathcal{O}_2^L$, except that for each equality check query, we compare the formal polynomials directly without evaluating them.

Additionally, we have the analogous sequence $\mathcal{O}_1^R, \mathcal{O}_2^R, \mathcal{O}_3^R$. We note here that X, Y, XY are linearly independent as formal polynomials over $\mathbb{Z}/p\mathbb{Z}$, and the group operation only allows us to add and subtract polynomials, so there is no way to distinguish whether we are given X, Y, XY or X, Y, Z. It follows that $\mathcal{O}_3^L$ and $\mathcal{O}_3^R$ are indistinguishable. By the same argument as in the previous example, after q queries, L_1 has $q+4$ elements, which are polynomials of degree at most 2 in the case of $\mathcal{O}_3^L$. Therefore,

$$\left|\Pr[\mathcal{A}^{\mathcal{O}_2^L(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_3^L(X,Y,Z)} = 0]\right| \leqslant \frac{(q+4)^2 2}{2p} = \frac{(q+4)^2}{p}$$

and similarly

$$\left|\Pr[\mathcal{A}^{\mathcal{O}_2^L(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_3^L(X,Y,Z)} = 0]\right| \leqslant \frac{(q+4)^2}{2p},$$

where the difference comes from the fact that the polynomials X, Y, Z are all of degree one, so L_1 will contain degree one polynomials in $\mathcal{O}_3^R$. Combining, we get

$$\begin{aligned}\left|\Pr[\mathcal{A}^{\mathcal{O}_1^L(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_1^R(X,Y,Z)} = 0]\right| \leqslant & \left|\Pr[\mathcal{A}^{\mathcal{O}_1^L(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_2^L(X,Y,Z)} = 0]\right| \\ & + \left|\Pr[\mathcal{A}^{\mathcal{O}_2^L(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_3^L(X,Y,Z)} = 0]\right| \\ & + \left|\Pr[\mathcal{A}^{\mathcal{O}_3^L(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_3^R(X,Y,Z)} = 0]\right| \\ & + \left|\Pr[\mathcal{A}^{\mathcal{O}_3^R(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_2^R(X,Y,Z)} = 0]\right| \\ & + \left|\Pr[\mathcal{A}^{\mathcal{O}_2^R(X,Y,Z)} = 0] - \Pr[\mathcal{A}^{\mathcal{O}_1^R(X,Y,Z)} = 0]\right| \\ \leqslant & \frac{(q+4)^2}{p} + \frac{(q+4)^2}{2p} \\ \leqslant & \frac{2(q+4)^2}{p}\end{aligned}$$

2.6 TYPES OF CRYPTOGRAPHIC ASSUMPTIONS

In a cryptographic assumption, some elements are usually sampled from a group and then handed over to an adversary. Such sampling of group elements is typically done in a restricted way which is captured by the following definition.

Definition 22. Assume we are given a group setting $\mathcal{GS} = (n, \mathcal{G}, I, \Phi, \mathcal{E})$, where $\mathbb{G}_i \in \mathcal{G}$ has generator P_i and the group operation is written additively. For a list of lists $\mathbf{L} = [L_1, \ldots, L_k]$ of Laurent polynomials in $(\mathbb{Z}/n\mathbb{Z})[X_1^{-1}, \ldots, X_m^{-1}, X_1, .., X_m]$, we define the distribution $\mathcal{D}_\mathbf{L}$ by the following procedure. Uniformly sample a point $\vec{x} \in (\mathbb{Z}/n\mathbb{Z})^m$ and return the list of lists $\mathbf{L}' = [L'_1, \ldots, L'_k]$ where $L'_i = [f_1(\vec{x})P_i, \ldots, f_{|L_i|}(\vec{x})P_i]$ for $f_j = L_i[j]$. A distribution $\mathcal{D}$ is *polynomially induced* if $\mathcal{D} = \mathcal{D}_{\mathbf{L}}$ for some $\mathbf{L}$.

As an example, the DDH assumption is defined by the single lists $L_l = [1, X, Y, XY]$ and $L_r = [1, X, Y, Z]$ for the left and right distribution, where the list entries are Laurent polynomials in $(\mathbb{Z}/n\mathbb{Z})[X^{-1}, Y^{-1}, Z^{-1}, X, Y, Z]$. These give two polynomially induced distributions $\mathcal{D}_{L_l}$ and $\mathcal{D}_{L_r}$ corresponding to the left and the right game.

Most hardness assumptions in generic group models belong to the following classes of decisional, computational, or generalized extraction problems stated with respect to a group setting $\mathcal{GS}$:

- Decisional problem for $\mathcal{D}_{\mathbf{L}}$ and $\mathcal{D}_{\mathbf{L}'}$:
  The adversary returns $b \in \{0, 1\}$ to distinguish the corresponding generic group models differing by their distributions.

- Computational problem for $\mathcal{D}_{\mathbf{L}}$, Laurent polynomial f, and group index i:
  We consider the event $L_i[h] = f(\vec{x})P_i$, where h is a handle returned by the adversary and $\vec{x}$ is the random point sampled by $\mathcal{D}_{\mathbf{L}}$.

- Generalized extraction problem for $\mathcal{D}_{\mathbf{L}}$, r, m, i_1, …, i_m and polynomials H_1, …, H_k and G_1, …, G_l: The adversary returns $\vec{a} \in (\mathbb{Z}/n\mathbb{Z})^r$ and handles $h_1, \ldots, h_m$. We consider the event that for all $i \in [k]$ and $j \in [l]$, the vector $\vec{x}$ sampled by $\mathcal{D}_{\mathbf{L}}$ and a vector $\vec{y}$ sampled by an oracle satisfies the constraints

  $$H_i(\vec{x}, \vec{y}, \vec{a}, \mathit{dlog}(L_{i_1}[h_1]), \ldots, \mathit{dlog}(L_{i_m}[h_m])) = 0$$

  and

  $$G_j(\vec{x}, \vec{y}, \vec{a}, \mathit{dlog}(L_{i_1}[h_1]), \ldots, \mathit{dlog}(L_{i_m}[h_m])) \neq 0.$$

  Here each H_i and G_j is a polynomial.

The above classification generalizes the one proposed by Maurer [Mau05]. Precisely, in addition to decisional and computational assumptions, Maurer considered "straight" extraction problems (such as discrete logarithm) in which the adversary has to extract the random value x of a handle. Our class of *generalized* *extraction problems* captures extraction problems like discrete logarithm, but also captures problems like the Strong Diffie-Hellman Problem [BB04]. Additionally, it is possible to express much more complicated assumptions, like security games for signature schemes. Moreover, note that our class of generalized extraction problems contains the class of computational problems.

Example 23. Set $r = 1$, $m = 0$, then $H(X, a_1) = X - a_1$ encodes DLOG as a generalized extraction problem and $r = m = 1$, $H(X, a_1, Z) = (X - a_1)Z - 1$ encodes SDH.

3 PROOF AUTOMATION IN THE GENERIC GROUP MODEL

A nice feature of the generic group model described in the previous chapter is that proving lower complexity bounds in most cases follows a template. This observation was first exploited by Boneh, Boyen and Goh [BBG05b] in their so called “Master Theorem”, which made it possible to prove the security of certain simple assumptions in the generic group model by checking an independence condition of polynomials.

In this chapter we develop principled, automated methods for analyzing hardness assumptions in generic group models. Our main contribution is essentially threefold. First, we reformulate master theorems in the style of the celebrated “computational soundness” theorem of Abadi and Rogaway [AR07], and formally show that the problem of analyzing assumptions in the generic group reduces to solving problems in polynomial algebra. Second, we systematically analyze these problems. While we show that the most general problem is undecidable, we distill a set of properties (capturing most interesting cases) for which the problem is decidable. Finally, by applying tools from linear algebra, we develop and implement automated procedures for verifying the conditions of master theorems, and thus the validity of hardness assumptions in generic group models. The concrete outcome of this work is an automated tool[1] which takes as input an assumption and outputs either a proof of its generic hardness or shows an algebraic attack against the assumption.

1 The tool is available at `http://www.easycrypt.info/GGA`

3.1 THE SYMBOLIC GROUP MODEL

The *symbolic group model* for a group setting $(n, \mathcal{G}, I, \Phi, \mathcal{E})$ and a polynomially-induced distribution $\mathcal{D}_\mathbf{L}$ provides the same adversary interface as the corresponding generic group model. The difference is that, internally, the challenger now stores lists of Laurent polynomials in $(\mathbb{Z}/n\mathbb{Z})[X_1^{-1}, \dots, X_m^{-1}, X_1, \dots, X_m]$ where $X_1, \dots, X_m$ are the variables occurring in $\mathbf{L}$. The oracles perform addition, negation, and equality checks in the polynomial ring. To define the polynomial operations corresponding to applications of isomorphisms and k-linear maps, observe that for all isomorphisms $\phi \in \Phi$ there is an $a \in (\mathbb{Z}/n\mathbb{Z})^\times$ such that $\phi(g_i) = g_j^a$. We therefore define the oracle $\mathrm{isom}_\phi(h)$ such that it computes $a \cdot L_i[h]$. Similarly, we define the oracle $\mathrm{map}_e(h_1, \dots, h_k)$ such that it computes $a \cdot (L_{i_1}[h_1] \cdots L_{i_k}[h_k])$. We also define symbolic versions $S(E)$ of events E used to define decisional, computational, and generalized extraction problems. For decisional problems and computational problems, the symbolic event is equal to the generic event, i.e., $S(E) = E$. For generalized extraction problems, the event E is translated by replacing all arguments of the Laurent polynomials H_j and G_j that are of the form *dlog*$(L_i[h])$ by $L_i[h]$. We denote the symbolic group model for a group setting $\mathcal{GS}$ and a distribution $\mathcal{D}_\mathbf{L}$ with $\mathrm{Sym}_{\mathcal{GS}}^{\mathcal{D}_\mathbf{L}}$ and the corresponding generic group model with $\mathrm{Gen}_{\mathcal{GS}}^{\mathcal{D}_\mathbf{L}}$.

We now specialize to the case, where the group order n in a group setting $(n, \mathcal{G}, I, \Phi, \mathcal{E})$ is prime. The composite-order case is treated separately in Section 3.6, since it's more involved and would unnecessarily clutter the presentation of the main ideas.

Theorem 24. *Let $(p, \mathcal{G}, I, \Phi, \mathcal{E})$ denote a group setting, $\mathcal{D}_\mathbf{L}$ a distribution, $\mathcal{A}$ an adversary performing at most q queries, and E the winning event of a decisional, computational, or generalized extraction problem. Let $d = d_1 + d_2$, where d_1 (resp. d_2) is an upper bound on the degree of the numerator (resp. denominator) of the Laurent polynomials occurring in the internal state of $\mathrm{Sym}_{\mathcal{GS}}^{\mathcal{D}_\mathbf{L}}(\mathcal{A})$ and $S(E)$, s the sum of the sizes of the lists in $\mathbf{L}$, and the event $S(E)$ contains at most e equality tests, then*

$$|\Pr[\ \mathrm{Gen}_{\mathcal{GS}}^{\mathcal{D}_\mathbf{L}}(\mathcal{A}) : E\ ] - \Pr[\ \mathrm{Sym}_{\mathcal{GS}}^{\mathcal{D}_\mathbf{L}}(\mathcal{A}) : S(E)\ ]| \leqslant \frac{(q+s)^2 d}{2p} + \frac{s d_2}{p} + \frac{e d}{p},$$

where the probability is taken over the coins of $\mathrm{Gen}_{\mathcal{GS}}^{\mathcal{D}_L}$ *and* $\mathcal{A}$*.*

Proof. An issue with Laurent polynomials in a polynomially induced distribution $\mathcal{D}_L$ is if we happen to sample values that are a root of a polynomial in a denominator, then the sampling will "fail". By Schwartz-Zippel, we know that this probability is bounded by sd_2/p.

First, observe that the number of equality checks between group elements (resp. polynomials) performed in $\mathrm{Gen}_{\mathcal{GS}}^{\mathcal{D}_L}$ (resp. $\mathrm{Sym}_{\mathcal{GS}}^{\mathcal{D}_L}$) is upper-bounded by $n = (q+s)^2/2 + e$. The adversary can perform at most $(q+s)^2/2$ distinct eq-queries since the lists $\mathbf{L}$ contain $q+s$ polynomials and the evaluation of the event requires e equality checks. To perform the proof, we utilize a hybrid argument that replaces equality checks $f_1/f_2 = g_1/g_2$ in $\mathrm{Sym}_{\mathcal{GS}}^{\mathcal{D}_L}$ by equality checks $f_1(\vec{x})g_2(\vec{x}) = g_1(\vec{x})f_2(\vec{x})$ for the random point $\vec{x}$ sampled by $\mathcal{D}_L$. The probability that the results for one equality check differ is then bounded by the Schwartz-Zippel Lemma, where the degree of $f_1g_2 - g_1f_2$ is bounded by d.

More formally, assuming sampling did not fail, for $i = 0, \ldots, n$, we obtain the hybrid game $\mathrm{Hyb}_{\mathcal{GS}}^{\mathcal{D}_L}\langle i\rangle$ and hybrid event $H\langle i\rangle(E)$ from $\mathrm{Sym}_{\mathcal{GS}}^{\mathcal{D}_L}$ and $S(E)$ by sampling the random point $\vec{x}$ on initialization and using $f_1/f_2 = g_1/g_2$ for equality checks 1 to i and $f_1(\vec{x})g_2(\vec{x}) = g_1(\vec{x})f_2(\vec{x})$ for equality checks $i+1$ to n. Then

$$\Pr[\ \mathrm{Gen}_{\mathcal{GS}}^{\mathcal{D}_L} : E\ ] = \Pr[\ \mathrm{Hyb}_{\mathcal{GS}}^{\mathcal{D}_L}\langle 0\rangle : H\langle 0\rangle(E)\ ]$$

and

$$\Pr[\ \mathrm{Hyb}_{\mathcal{GS}}^{\mathcal{D}_L}\langle n\rangle : H\langle n\rangle(E)\ ] = \Pr[\ \mathrm{Sym}_{\mathcal{GS}}^{\mathcal{D}_L} : S(E)\ ].$$

To complete the proof, we use the Schwartz-Zippel Lemma to prove that for all $i \in [n]$,

$$|\Pr[\ \mathrm{Hyb}_{\mathcal{GS}}^{\mathcal{D}_L}\langle i-1\rangle(\mathcal{A}) : H\langle i-1\rangle(E)\ ] - \Pr[\ \mathrm{Hyb}_{\mathcal{GS}}^{\mathcal{D}_L}\langle i\rangle(\mathcal{A}) : H\langle i\rangle(E)\ ]| \leqslant d/p.$$

The two games are equivalent unless the i-th equality-check returns different results in the left and the right experiment. This is equivalent to $f_1g_2 - g_1f_2 \neq 0$ and $f_1(\vec{x})g_2(\vec{x}) - g_1(\vec{x})f_2(\vec{x}) = 0$ where $f_1g_2 - g_1f_2$ is a polynomial of degree at most d and $\vec{x}$ is uniform independent of f_1/f_2 and g_1/g_2 since earlier equality checks use equality on polynomials in both experiments. $\square$

By applying this theorem, we can therefore analyze the hardness of assumptions in the simpler symbolic model. We note that existing master theorems usually include a similar step in their proofs. Here we explicitly prove the equivalence of the Gen and Sym experiments. This stronger result is required for our decidability results.

3.2 OUR MASTER THEOREM

In this section we state our master theorem for decisional, non-interactive problems. In Section 3.5, we give a master theorem for interactive assumptions which cover generalized extraction problems (and computational ones per Section 2.6).

To state our theorem, we first define the completion $\mathcal{C}(\mathbf{L})$ of a list $\mathbf{L}$ with respect to the group setting $(p, \mathcal{G}, I, \Phi, \mathcal{E})$. This notion will be instrumental to define the side condition of our master theorem. Intuitively speaking, given a list $\mathbf{L}$, its completion $\mathcal{C}(\mathbf{L})$ is the list of all Laurent polynomials that can be computed by the adversary by applying isomorphisms and maps to Laurent polynomials in $\mathbf{L}$.

We compute the completion $\mathcal{C}(\mathbf{L})$ of $\mathbf{L}$ in two steps. In the first step, we compute the *recipe lists* $\{R_i\}_{i \in I}$ using the algorithm given in Figure 1. The elements of the recipe lists are monomials over the variables $W_{i,j}$ for $(i, j) \in I \times [|L_i|]$. The monomials characterize which products of elements in $\mathbf{L}$ the adversary can compute by applying isomorpisms and maps. The result of the first step is independent of the elements in the lists $\mathbf{L}$ and only depends on the lengths of the lists. In the second step, we compute the actual Laurent polynomials from the recipes as

$$\mathcal{C}(\mathbf{L})_i = [m_1(\mathbf{L}), \ldots, m_{|R_i|}(\mathbf{L})] \text{ for } [m_1, \ldots, m_{|R_i|}] = R_i,$$

where every m_i is a monomial over the variables $W_{i,j}$ and $m_i(\mathbf{L})$ denotes the result of evaluating the monomial m_i for the values $L_i[j_i]$.

To ensure that the computation of the recipes terminates, we restrict ourselves to group settings without cycles. We also assume that the group setting contains a target group. Formally, for a group setting $(p, \mathcal{G}, I, \Phi, \mathcal{E})$, we define the weighted directed graph $G = (V, E)$ with $V = \mathcal{G}$ and E defined as follows. For each isomorphism $\mathbb{G}_i \to \mathbb{G}_j \in \Phi$, there is an edge from $\mathbb{G}_i$ to $\mathbb{G}_j$ of weight 0. Similarly, given

```
foreach $i \in I$ : $S'_i = \emptyset$; $S_i = \{W_{i,1}, \dots, W_{i,|L_i|}\}$
while $\vec{S} \neq \vec{S'}$ :
    $\vec{S'} := \vec{S}$
    foreach $e : \mathbb{G}_{j_1} \times \dots \times \mathbb{G}_{j_n} \to \mathbb{G}_{j_{n+1}} \in \mathcal{E}$ :
        $S_{j_{n+1}} := S_{j_{n+1}} \cup \{f_1 \cdots f_n \mid f_i \in S_{j_i},\ i \in [n]\}$
    foreach $\phi : \mathbb{G}_i \to \mathbb{G}_j \in \Phi$ : $S_j := S_j \cup S_i$
foreach $i \in I$ : $R_i := \mathit{setToList}(S_i)$
```

FIGURE 1: Computation of lists of recipes R_i for input lists L_i.

any $\mathbb{G}_{i_1} \times \dots \times \mathbb{G}_{i_n} \to \mathbb{G}_{i_{n+1}} \in \mathcal{E}$, there are edges from $\mathbb{G}_{i_j}$ to $\mathbb{G}_{i_{n+1}}$ of weight 1 for $j \in [n]$. We assume that the graph G contains no loops of *positive* weight. Furthermore, we assume there is a unique $\mathbb{G}_t \in V$ called the *target group*, such that from any $\mathbb{G}_i \in V$ there is a path to $\mathbb{G}_t$ and $\mathbb{G}_t$ does not have any outgoing edges, i.e. it's a sink node of the graph that is reachable from all other vertices.

Example 25. Assume we are given a leveled 3-linear map with inputs $L_1 = [1, X]$ and $L_i = \emptyset$ for $i = 2, 3, 4$. Then $S_1 = \{W_{1,1}, W_{1,2}\}$ and $S_i = \emptyset$ for $i = 2, 3, 4$. Since there are no isomorphisms, the body of the *while* loop reduces to the first *for* loop and to checking whether or not we have saturated the sets. The completion algorithm proceeds as follows (writing down the sets that have changed):

1. $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ gives $S_2 = \{W_{1,1}^2, W_{1,1}W_{1,2}, W_{1,2}^2\}$.

2. $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ and $e : \mathbb{G}_2 \times \mathbb{G}_2 \to \mathbb{G}_4$ gives:

$$S_3 = \{fg \mid f \in S_1,\ g \in S_2\} = \{W_{1,1}^3, W_{1,1}^2 W_{1,2}, W_{1,1}W_{1,2}^2, W_{1,2}^3\}$$
$$S_4 = \{fg \mid f \in S_2,\ g \in S_2\} = \{W_{1,1}^4, W_{1,1}^3 W_{1,2}, W_{1,1}^2 W_{1,2}^2, W_{1,1}W_{1,2}^3, W_{1,2}^4\}$$

3. At the last iteration we union $\{fg \mid f \in S_1,\ g \in S_3\}$ to S_4. It turns out that this adds no new elements to S_4.

Finally, we plug in $W_{1,1} = 1$ and $W_{1,2} = X$ to get the completions (using the ordering in which the elements were written down above):

$$L_1 = [1, X],\ L_2 = [1, X, X^2],\ L_3 = [1, X, X^2, X^3],\ L_4 = [1, X, X^2, X^3, X^4]$$

Theorem 26. *Let* $\mathcal{GS} = (p, \{\mathbb{G}_i\}_{i\in I}, I, \Phi, \mathcal{E})$ *denote a group setting, and* $\mathcal{D}_{\mathbf{L}}, \mathcal{D}_{\mathbf{L}'}$ *be polynomially-induced distributions such that* $|L_i| = |L'_i|$ *for all* $i \in I$*. Let* t *denote the index of the target group,* $s = \sum_{i\in I} |L_i|$, $r = |\mathcal{C}(\mathbf{L})_t|$, *and let* $d = d_1 + d_2$, *where* d_1 *(resp.* d_2*) denotes an upper bound for the total degrees of the polynomials in the numerator (resp. denominator) of the Laurent polynomials in the completion of the lists. If*

$$\{\vec{a} \in \mathbb{F}_p^r \mid \vec{a} \cdot \mathcal{C}(\mathbf{L})_t = 0\} = \{\vec{a} \in \mathbb{F}_p^r \mid \vec{a} \cdot \mathcal{C}(\mathbf{L}')_t = 0\},$$

then

$$|\Pr[\,\mathrm{Gen}^{\mathcal{GS}}_{\mathcal{D}_{\mathbf{L}}}(\mathcal{A}) = 1\,] - \Pr[\,\mathrm{Gen}^{\mathcal{GS}}_{\mathcal{D}_{\mathbf{L}'}}(\mathcal{A}) = 1\,]| \leqslant \frac{(s+q)^2 d}{p} + \frac{2sd_2}{p}$$

for all adversaries $\mathcal{A}$ *that perform at most* q *operations.*

Proof. By using a standard hybrid argument we can apply our Theorem 24 to switch both experiments $\mathrm{Gen}^{\mathcal{D}_{\mathbf{L}}}_{\mathcal{GS}}(\mathcal{A})$ and $\mathrm{Gen}^{\mathcal{D}_{\mathbf{L}'}}_{\mathcal{GS}}(\mathcal{A})$ from the generic group model to the symbolic group model. Hence, we obtain:

$$|\Pr[\,\mathrm{Gen}^{\mathcal{D}_{\mathbf{L}}}_{\mathcal{GS}}(\mathcal{A}) = 1\,] - \Pr[\,\mathrm{Sym}^{\mathcal{D}_{\mathbf{L}}}_{\mathcal{GS}}(\mathcal{A}) = 1\,]| \leqslant \frac{(s+q_o)^2 d}{2p} + \frac{sd_2}{p}$$
$$|\Pr[\,\mathrm{Sym}^{\mathcal{D}_{\mathbf{L}'}}_{\mathcal{GS}}(\mathcal{A}) = 1\,] - \Pr[\,\mathrm{Gen}^{\mathcal{D}_{\mathbf{L}'}}_{\mathcal{GS}}(\mathcal{A}) = 1\,]| \leqslant \frac{(s+q_o)^2 d}{2p} + \frac{sd_2}{p}$$

This leaves us with bounding

$$|\Pr[\,\mathrm{Sym}^{\mathcal{D}_{\mathbf{L}}}_{\mathcal{GS}}(\mathcal{A}) = 1\,] - \Pr[\,\mathrm{Sym}^{\mathcal{D}_{\mathbf{L}'}}_{\mathcal{GS}}(\mathcal{A}) = 1\,]|$$

However, note that the view of the adversary $\mathcal{A}$ in this symbolic game essentially depends on the equality queries that are performed on all polynomials computed by $\mathcal{A}$ and stored by the challenger. By viewing the completion $\mathcal{C}(\mathbf{L})$ as the generating set of a vector space V (which is all polynomials computable by the adversary starting from $\mathbf{L}$), then $\{\vec{a} \in \mathbb{F}_p^r \mid \vec{a} \cdot \mathcal{C}(\mathbf{L})_t = 0\}$ is the kernel of the map that expresses every polynomial in V as a linear combination of polynomials in $\mathcal{C}(\mathbf{L})$. In particular, note that such map also encodes equalities of polynomials in V. Hence, the validity of our side condition essentially says that the equalities seen by the adversary in both games (i.e., w.r.t. lists $\mathbf{L}$ and $\mathbf{L}'$) are the same. Namely, the adversary gets an identical view in the two experiments. □

Note that deciding the side condition is sufficient for deciding the hardness of the corresponding decisional problem for a fixed group setting and fixed distributions. Either the side condition is satisfied or there exists an $\vec{a} \in \mathbb{F}_p^r$ that is included in one of the sets, but not in the other one. In the first case, the distinguishing advantage is upper-bounded by the ϵ given above. In the second case, we can construct an adversary that distinguishes the two symbolic models with probability 1, which implies that it distinguishes the corresponding generic models with probability $1 - \epsilon$. Note that for real-or-random assumptions where the adversary is given $\hat{\mathbf{L}}$ and must distinguish f from a fresh variable Z in the target group $\mathbb{G}_t$, our side condition simplifies to $\sum_{j=1}^{r} a_j \mathcal{C}(\hat{\mathbf{L}})_t[j] \neq f$ for all $\vec{a} \in \mathbb{F}_p^r$. This is similar to the independence condition in the BBG master theorem [BBG05b].

3.3 NON-PARAMETRIC ASSUMPTIONS

In this section, we present methods to automatically verify or falsify the hardness of decisional assumptions. As mentioned earlier, our master theorem is stated with respect to a fixed group setting and fixed distributions. To consider multiple group settings or distributions at once, we define a decisional assumption $\mathbb{A}$ as a possibly infinite set of triples $(\mathcal{GS}, \mathcal{D}_{\mathbf{L}}, \mathcal{D}_{\mathbf{L}'})$. $\mathbb{A}$ is *generically hard* if the distinguishing probability is upper-bounded by ϵ in Theorem 26 for all triples in $\mathbb{A}$.

An assumption is non-parametric if only the concrete groups, isomorphisms, and n-linear maps vary, but the structure of the group setting and the lists $\mathbf{L}$ and $\mathbf{L}'$ defining the distributions remain fixed. This captures assumptions such as "3-lin is hard in all groups with a symmetric 3-linear map". Conversely, an assumption is parametric if one or more of these restrictions do not hold.

We perform the following computations over $\mathbb{Z}$ to decide the hardness of a decisional assumption defined by lists $\mathbf{L}$ and $\mathbf{L}'$ for all group settings $\mathcal{GS}$ with a given index set and types of isomorphisms and n-linear maps.

1. Initialize the set T of distinguishing tests and the set E of exceptional primes to the empty set $\emptyset$.

2. Compute the completions $\mathcal{C}(\mathbf{L})$ and $\mathcal{C}(\mathbf{L}')$ and set $\overline{L}_t := \mathcal{C}(\mathbf{L})_t$, $\overline{L'}_t := \mathcal{C}(\mathbf{L}')_t$

3. Compute a generating set K of the $\mathbb{Z}$-module $\{\vec{a} \in \mathbb{Z}^{|\overline{L}_t|} \mid \vec{a} \cdot \overline{L}_t = 0\}$ as follows:
    a) Represent all polynomials $g \in \overline{L}_t$ as vectors $\vec{v_1}, \dots, \vec{v_n}$ and denote by M the matrix, where row i is $\vec{v_i}$ with respect to the basis *monomials*$(\overline{L}_t)$.
    b) Compute the Hermite Normal Form N of M and read off a generating set K of the left kernel from N and the transformation matrix. Set $E := E \cup F$ where F is the set of factors of pivots of N.

    Perform the same steps for $\overline{L'}_t$ to obtain M' and K'.
4. Check for every $\vec{k} \in K$ if $\vec{k}M' = 0$. If $\vec{k}M' = \vec{c} \neq 0$, then set $T := T \cup \vec{k}$ and $E := E \cup F$ where F denotes the set of common prime factors of the components of $\vec{c}$. Perform the same steps for K' and M.
5. Compute distinguishing probability ϵ from degrees in $\overline{L}_t$ and $\overline{L}'_t$.
6. If T is empty, return that distinguishing probability is upper-bounded by ϵ except (possibly) for primes in E. If T is nonempty, return that using the tests in T, an adversary can distinguish with probability $1 - \epsilon$ except (possibly) for primes in E.

Note that performing division-free computations over $\mathbb{Z}$ allows us to track the set of exceptional primes, which we return. We have implemented this algorithm in a tool that takes a group setting and two sequences of group elements as input and decides if the corresponding decisional assumption is hard, returning ϵ, E, and the distinguishing tests T (if nonempty).

Example 27. We show that the following assumption is insecure in a bilinear group using our method. This is just a slight modification of DDH where not all inputs are monomials, since it better highlights how the algorithm works. Let $L_1 = [1, X, Y, XY + Y]$ or $L'_1 = [1, X, Y, Z]$ with $L_2 = L'_2 = \emptyset$, with $\mathbf{L} = [L_1, L_2]$ and $\mathbf{L}' = [L'_1, L'_2]$. Using the notation above, we have that

$$\overline{L}_2 = [1, X, Y, XY + Y, X^2, XY, X^2Y + XY, Y^2, XY^2 + Y^2, X^2Y^2 + 2XY^2 + Y^2]$$
$$\overline{L}'_2 = [1, X, Y, Z, X^2, XY, XZ, Y^2, YZ, Z^2]$$

To compute the left kernels, we choose the following monomial bases:

$$\begin{aligned}\mathbb{B} &= (1, X, Y, XY, X^2, Y^2, XY^2, X^2Y^2)\\ \mathbb{B}' &= (1, X, Y, Z, X^2, XY, XZ, Y^2, YZ, Z^2)\end{aligned}$$

We obtain the following matrices, where each row represents an element in the completions in the corresponding monomial basis.

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 1 \end{bmatrix}, \; M' = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

M' is the identity matrix, so it's already in HNF and has trivial kernel. The HNF for M and its transformation matrix K are

$$N = \begin{bmatrix} 1&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&0&0&0 \end{bmatrix}, K = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&-1&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&-1&1&0&0&0\\ 0&0&0&0&0&0&0&-1&1&0\\ 0&0&0&0&0&0&0&1&-2&1\\ 0&0&1&-1&0&1&0&0&0&0 \end{bmatrix}$$

The generators of the kernel are the rows of the matrix K corresponding to the zero rows of the HNF, N, of M. All the pivots of N are 1, so there are no exceptional primes. Thus the kernel is generated by $\vec{a} = (0,0,1,-1,0,1,0,0,0,0)$ and $\vec{a}$ denotes the relation $L_1[3] - L_1[4] + e(L_1[2], L_1[3]) = 0$. We also have that $\vec{a}M' = \vec{a} \neq \vec{0}$, so this relation is not present in $\overline{L}_2'$. Furthermore, $\vec{a}$ is not zero modulo any prime, so there are no exceptional primes. In addition to what is shown here, our software keeps track of how each element of $\overline{L}_2$ was constructed, so it can print out the formulas for all algebraic attacks against the assumption.

3.4 PARAMETRIC ASSUMPTIONS

For parametric decisional assumptions, we restrict ourselves to the real-or-random case. The approach can also be adapted to handle computational assumptions. We also assume that input distributions are defined by polynomials instead of Laurent polynomials. Our methods do extend to Laurent polynomials with some minor modifications, but no current parametric assumptions known to us require this level of generality. We distinguish parametricity in two dimensions. First, an assumption may be parameterized by *range limits* $l_1, \dots, l_m$

(ranging over $\mathbb{N}$) that determine the size of the adversary input. We use *range expressions* $\forall r \in [\alpha, \beta].\, h_r$, where α and β are polynomials over range limits, to express such assumptions. The polynomials h_r can use the *range index* r in the exponent or as the index of an indexed variable X_r. We will denote range expressions with capital letters R. Second, the group setting of an assumption may be parameterized by an *arity* k that captures the maximum number of multiplications that can be performed.

Parametricity in the input size allows us to analyze assumptions such as "l-DHE is hard for all l". Parametricity in the arity allows us to analyze assumptions such "2-BDH is hard for all k-linear groups". Combining both types of parametricity allows us to analyze assumptions such as "k-lin is hard in k-linear groups" or "(l, k)-MMDHE is hard for all l and $k \geqslant 3$". In the following, we will present two methods that deal with both parametricity in the input size and parametricity in the arity. The first method assumes a fixed number of random variables. The second method allows for indexed random variables, but assumes that the degree of adversary input and challenge is fixed.

The way we handle parametric assumptions is based on a trivial observation. Let $S_1, \dots, S_n$ be not necessarily distinct sets of monomials. Then a polynomial f belongs to the span of the product set $S_1 \cdots S_n$ if and only if all its terms belong to $S_1 \cdots S_n$. Therefore, we need to define products of range expressions as well as methods to check whether a monomial is an instance of a range expression. Additionally, we must assume that all inputs are monomials.

Fixed Number of Variables.

We assume a real-or-random decisional assumption in a (leveled) k-linear group where the challenge polynomial g is in the target group, and the adversary input is expressed using range expressions $R_1, \dots, R_n$ on the levels $\lambda_1, \dots, \lambda_n$. Here λ_i is either of the form c or of the form $k - c$ for a constant $c \in \mathbb{N}$. Furthermore, we assume that the assumption uses random variables $\vec{X}$ and range limits $\vec{l}$. To simplify the presentation, we will use the notation $\vec{X}^{\vec{f}} = X_1^{f_1} \cdots X_m^{f_m}$. Then the ranges are of the form

$$R_i = \forall r_{i,1} \in [\alpha_{i,1}, \beta_{i,1}], \dots, r_{i,t_i} \in [\alpha_{i,t_i}, \beta_{i,t_i}].\, \vec{X}^{\vec{f_i}}$$

where every $\alpha_{i,j}$ and $\beta_{i,j}$ is a polynomial over $\vec{l}$ and every $f \in \vec{f_i}$ is a polynomial over k, $\vec{l}$, and $r_{i,1}, \ldots, r_{i,t_i}$. The challenge polynomial is of the form $g = \sum_{i=1}^{w} c_i \vec{X}^{\vec{u_i}}$. Using the independence condition derived from Theorem 26, it follows that real distribution and the random distribution are indistinguishable iff there is a monomial $\vec{X}^{\vec{u_i}}$ that is not an element of the completion of the R_i.

To check this condition, we proceed in two steps. In the first step, we compute a single range expression $\overline{R}$ that denotes the completion of the R_i in the target group. In the second step, we check for each $\vec{X}^{\vec{u_i}}$ whether $\vec{X}^{\vec{u_i}} \in \overline{R}$, by encoding the required equalities of the exponent-polynomials into a set of diophantine (in)equalities. We then show that satisfiability checking for such constraints is undecidable in general. Nevertheless, we identify two decidable fragments and demonstrate that SMT solvers can handle most instances derived from practical cryptographic assumptions, even those that are not in the decidable fragments.

If $R_1, \ldots, R_n$ denote the sets $S_1, \ldots, S_n$, then the completion $\overline{R}$ of $R_1, \ldots, R_n$ in the target group must denote the set

$$\bigcup_{\vec{\delta} \in \mathbb{N}^n \text{ s.t. } \sum_{i=1}^{n} \delta_i \cdot \lambda_i = k} S_1^{\delta_1} \cdots S_n^{\delta_n}$$

where $SS' = \{ss' \mid s \in S \wedge s' \in S'\}$ and $S^{\delta} = \{\prod_{i=1}^{\delta} s_i | s_1 \in S \wedge \ldots \wedge s_\delta \in S\}$. We therefore define multiplication of range expressions with distinct range indices as

$$\begin{aligned} &(\forall r_1 \in [\alpha_1, \beta_1], \ldots, r_t \in [\alpha_t, \beta_t].\, \vec{X}^{\vec{f}})(\forall r'_1 \in [\alpha'_1, \beta'_1], \ldots, r'_s \in [\alpha'_{t'}, \beta'_{t'}].\, \vec{X}^{\vec{f}'}) \\ &= \forall r_1 \in [\alpha_1, \beta_1], \ldots, r_t \in [\alpha_t, \beta_t], r'_1 \in [\alpha'_1, \beta'_1], \ldots, r'_s \in [\alpha'_{t'}, \beta'_{t'}].\, \vec{X}^{\vec{f}+\vec{f}'}. \end{aligned}$$

To define the δ-fold product of a range expression, we restrict ourselves to exponent-polynomials that can be expressed as $f + g$ such that $f = \sum_{j=1}^{t} r_j \, \phi_j(\vec{l}, k)$ for polynomials ϕ_j in $\mathbb{Z}[\vec{l}, k]$ and such that g is a polynomial in $\mathbb{Z}[\vec{l}, k]$. The δ-fold product is then defined as

$$\begin{aligned} &(\forall r_1 \in [\alpha_1, \beta_1], \ldots, r_m \in [\alpha_t, \beta_t].\, \vec{X}^{\vec{f}+\vec{g}})^{\delta} \\ &= \forall r_1 \in [\delta\alpha_1, \delta\beta_1], \ldots, r_m \in [\delta\alpha_t, \delta\beta_t].\, \vec{X}^{\vec{f}+\delta\vec{g}}. \end{aligned}$$

Given range expressions $R_1, \ldots, R_n$, we can now compute $\overline{R}$ by introducing fresh variables $\delta_1, \ldots, \delta_n$, computing the range expressions $R_i^{\delta_i}$, and then computing the product of these range expressions.

The remaining task is now to check if

$$\vec{X}^{\vec{u}} \in (\forall r_1 \in [\alpha_1, \beta_1], \ldots, r_t \in [\alpha_t, \beta_t]. \vec{X}^{\vec{f}}) = \overline{R}$$

where $\vec{u} \in \mathbb{Z}[\vec{l}, k]^m$, $\alpha_i, \beta_i \in \mathbb{Z}[\vec{\delta}, \vec{l}]$, $\vec{f} \in \mathbb{Z}[\vec{l}, k, r_1, \ldots, r_t]^m$, and $\sum_{i=1}^n \delta_i \cdot \lambda_i = k$. To achieve this, we compute the following set of integer constraints that is satisfiable iff $\vec{X}^{\vec{u}} \in \overline{R}$:

$$\begin{cases} 0 \leqslant \delta_i & \text{for } i \in [1, n] \\ \alpha_i \leqslant r_i \leqslant \beta_i & \text{for } i \in [1, t] \\ u_i = f_i, & \text{for } i \in [1, m] \\ \sum_{i=1}^n \delta_i \lambda_i = k \end{cases}$$

If we allow for both types of parametricity, it is possible to reduce Hilbert's 10th problem to the generic hardness of cryptographic assumptions expressed as previously described. This yields the following theorem.

Theorem 28. *Deciding hardness of parametric assumptions with a fixed number of variables in the generic group model is undecidable, even if all exponent-polynomials are linear in range limits, range indices, and the arity.*

Proof. Given a Diophantine problem, by introducing new variables, we can always reduce the equation to a system of the following form

$$\begin{cases} c_1 X_1 + \ldots + c_n X_n = d \\ X_i = X_j \cdot X_k \end{cases} . \tag{1}$$

We show that this system can always be encoded in the set of equations

$$\begin{cases} 0 \leqslant \delta_i & \text{for } i \in [1, n] \\ \alpha_i \leqslant r_i \leqslant \beta_i & \text{for } i \in [1, t] \\ u_i = f_i, & \text{for } i \in [1, m] \\ \sum_{i=1}^n \delta_i \lambda_i = k \end{cases}$$

by appropriate choice of decision function and input polynomials. Undecidability then follows from Matiyasevich's theorem [Mat70]. We note that for undecidability, it's enough to look for solutions over the naturals, since the general case over the integers reduces to this by independently replacing each variable X by $\pm X$ and checking all the possible equations over the naturals.

We now show how to encode the system (1) as a range and arity parametric assumption with all inputs at level one and the decision polynomial a monomial at the target level. First, we look at the linear relation $c_1X_1 + \ldots + c_nX_n = d$. Let $l_1, \ldots, l_n$ be fresh range limit variables. We add the input monomial

$$M = Z_1^{c_1 l_1 + \ldots + c_n l_n} Z_2$$

as input, where Z_1, Z_2 are freshly introduced variables not used in any other input monomial. Conversely, we add to the decision polynomial f the factors

$$Z_1^d Z_2.$$

The degree equations corresponding to the variables Z_1 and Z_2 are now

$$\begin{cases} d = \delta(c_1 l_1 + \ldots + c_n l_n) \\ 1 = \delta \end{cases},$$

which encode $d = c_1 l_1 + \ldots + c_n l_n$. Having switched variables in (1) to range limits, we look at an equation $l_i = l_j \cdot l_m$. Again, we let Z_1, Z_2 be new fresh variables and add the input monomial $M = Z_1 Z_2^{l_m}$. Conversely, we multiply f with the factors $Z_1^{l_j} Z_2^{l_i}$. The degree equations now become

$$\begin{cases} l_j = \delta \\ l_i = \delta \cdot l_m \end{cases},$$

which gives us precisely $l_i = l_j \cdot l_m$. Since k is not used anywhere in the input or f it only appears in the first equation in the system (1). Therefore, k can grow unbounded, so $\sum_{i=1}^{t} \delta_i \leqslant k$ adds no restrictions on the δ_i except what was forced by our computations above. Similarly, range indices are never used, so no limitations are forced on the range limits except that they are positive. Therefore, we have encoded the system (1) under the sole requirement that $l_i \in \mathbb{N}$ for each i while only placing restrictions on the other variables that can be trivially independently satisfied. □

However, for a restricted class of assumptions, the problem is decidable.

Theorem 29. *For all parametric assumptions with a fixed number of variables such that all exponent-polynomials $f_{i,j}$ and range bounds $\alpha_{i,j}$ and $\beta_{i,j}$ in the input are linear, and either (1) the arity k is fixed or (2) the assumption does not contain range limits l_i and the input exponent-polynomials do not use k, deciding hardness in the generic group model is decidable.*

Sketch. In both cases, we transform the constraint system into a system of linear constraints. Note that the first type of constraint is already linear. In the first case, the arity k is fixed and we can eliminate the variables δ_i by performing a case distinction since there are only finitely many possible values. Then, the constraints of the first and fourth type are constant and the constraints of the second and third type are linear. If there are no range limits, then the range bounds are constants and we can eliminate the range indices by expanding all range expressions into finite sets of monomials. Then the constraints of the second type are constant and we can linearize the constraints of the last type using Proposition 30. For constraints of the third type, every u_i is a linear polynomial in $\mathbb{Z}[k]$ and every f_i is a linear polynomial in $\mathbb{Z}[\vec{\delta}, k]$. □

Proposition 30. *The equation $\sum_{i=1}^{t} \delta_i \lambda_i = k$ can always be split into a finite number of cases, where the remaining equation is linear.*

Proof. The variables λ_i are by definition either of the form c_i or $k - c_i$, where c_i is a constant. In the former case $\delta_i c_i$ is a first-degree term. In the latter case $\delta_i(k - c_i)$ is a degree two term. By arranging terms, we may write the equation in the form

$$\sum_{i=1}^{r} \delta_i c_i + \sum_{i=r+1}^{t} \delta_i (k - c_i) = k.$$

Write $c = \max\{c_i \mid i = r+1, \dots, t\}$, so that

$$(k-c)\sum_{i=r+1}^{t} \delta_i = \sum_{i=r+1} \delta_i(k-c) \leqslant \sum_{i=r+1}^{t} \delta_i(k-c_i) \leqslant k$$

and therefore

$$\sum_{i=r+1}^{t} \delta_i \leqslant \frac{k}{k-c} \leqslant \frac{c+1}{c+1-c} = c+1.$$

This shows that there are a finite number of possible values for $\delta_{r+1}, \dots, \delta_t$. This leads to at most $\binom{c+t-r}{t-r-1}$ possible choices for $\delta_{r+1}, \dots, \delta_t$. □

We have implemented this method in our tool and use Z3 [DMB08] to check the constraints. Our experiments confirm that Z3 can prove most assumptions taken from the literature, even those outside the decidable fragment.

Example 31. The decisional l-BDHE [BGW05] problem is as follows: In a symmetric bilinear group, $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$, we are given the input $(h, g, g^x, g^{x^2}, \dots, g^{x^l}, g^{x^{l+2}}, \dots, g^{x^{2l}})$. Writing $h = g^y$, the problem is to distinguish $e(h,g)^{x^{l+1}} = e(g,g)^{yx^{l+1}}$ from random. Symbolically, the problem is given as the input

$$M_1 = 1,\ M_2 = Y,\ M_3 = \forall i \in [0,l].\, X^i,\ M_4 = \forall j \in [0, l-2].\, X^{l+2+j},\ g = YX^{l+1},$$

where the M_i are all at level 1 with the challenge at level 2. Therefore, we compute

$$\begin{aligned}
M_1^{\delta_1} &= 1\\
M_2^{\delta_2} &= Y^{\delta_2}\\
M_3^{\delta_3} &= \forall i \in [0, \delta_3 l].\, X^i\\
M_4^{\delta_4} &= \forall j \in [0, \delta_4(l-2)].\, X^{j+\delta_4(l+2)}\\
M_1^{\delta_1}M_2^{\delta_2}M_3^{\delta_3}M_4^{\delta_4} &= \forall i \in [0, \delta_3 l],\ j \in [0, \delta_4(l-2)].\, X^{i+j+\delta_4(l+2)}Y^{\delta_2}
\end{aligned}$$

This yields the following set of constraints

$$\begin{cases} 0 \leqslant \delta_1, \delta_2, \delta_3, \delta_4 \\ 0 \leqslant i \leqslant \delta_3 l \\ 0 \leqslant j \leqslant \delta_4(l-2) \\ l+1 = i+j+\delta_4(l+2) \\ 1 = \delta_2 \\ \delta_1+\delta_2+\delta_3+\delta_4 = 2 \end{cases} .$$

The fourth and (resp. fifth) equation corresponds to taking the degrees of both sides of the equation $YX^{l+1} = M_1^{\delta_1} M_2^{\delta_2} M_3^{\delta_3} M_4^{\delta_4}$ with respect to X (resp. Y). In this case the system trivially linearizes, since there are only finitely many possible values for the δ_i, $i = 1, \ldots, 4$.

3.5 INTERACTIVE ASSUMPTIONS

In this section, we present our methods for the analysis of interactive assumptions such as LRSW [LRSW99]. We focus on assumptions where exactly *one* additional oracle $\mathcal{O}$ is provided to the adversary and the problem is a generalized extraction problem. In the remainder, we fix a group setting $\mathcal{GS} = (p, \{\mathbb{G}\}_{i \in I}, I, \Phi, \mathcal{E})$ and a distribution $\mathcal{D}_{\mathbf{L}}$. We use $\vec{X}$ to denote the variables occurring in $\mathbf{L}$ and $\vec{x}$ to denote the point sampled by $\mathcal{D}_{\mathbf{L}}$.

Generalizing GGM *and* SGM*.*

Our first step is generalizing the generic group and symbolic group models to the interactive setting. In order to do this, we need to give precise definitions of our oracles that we can mathematically work with. We first note that an oracle operates as follows:

1. It takes as *parameters* elements in $\mathbb{F}_p$ as well as handles to group elements.

2. It returns handles to *group elements* that are formed from the parameters, values randomly sampled by $\mathcal{D}_L$ and finally elements in $\mathbb{F}_p$ randomly sampled by the oracle.

In the symbolic model, the oracle has the same interface, except that when internally sampling a new value, it instead creates a new formal variable. Instead of handles to group elements, it then returns handles to formal Laurent polynomials in the ring augmented with the new formal variables.

Denote parameters in $\mathbb{F}_p$ given to the oracle using the variables $A_1, \dots, A_n$, the parameters in various $\mathbb{G}_i$ by $Z_1, \dots, Z_r$. Furthermore, denote the elements in $\mathbb{F}_p$ internally sampled by the oracle by the variables $Y_1, \dots, Y_m$. An oracle can then be described as a vector of some length k of Laurent polynomials in the ring

$$\mathbb{F}_p[\vec{X}, \vec{X}^{-1}, \vec{A}, \vec{A}^{-1}, \vec{Y}, \vec{Y}^{-1}, \vec{Z}, \vec{Z}^{-1}]$$

together with a vector of length k of elements in I. The latter vector denotes which groups the corresponding polynomials returns handles to. Here we use the notation $\vec{Y} = (Y_1, \dots, Y_m)$, where we use $\vec{Y}^{-1}$ to denote the vector $(Y_1^{-1}, \dots, Y_m^{-1})$. In the symbolic model, instead of sampling values for the Y_i, the oracle instead creates fresh new variables for each Y_i and substitutes them for each corresponding Y_i in the Laurent polynomials returned. The oracle then returns handles to the corresponding formal Laurent polynomials appended to their corresponding groups.

Finally, we augment the oracle definition with a nonnegative integer value q, which is a limit for the number of queries of the oracle that is allowed.

Example 32. A signing oracle for the randomizable structure-preserving signature scheme in Figure 5 in Section 4.5.4 can be described as the following vector of Laurent polynomials

$$(Y_1, (Z_1 X_1 + X_2)/Y_1)$$

in the ring $\mathbb{F}_p[X_1, X_2, X_1^{-1}, X_2^{-1}, Y_1, Y_1^{-1}, Z_1, Z_1^{-1}]$, together with the vector $(2, 2)$. The latter vector comes from the fact that the oracle returns two elements in $\mathbb{G}_2$. In the notation of Figure 5 in Section 4.5.4, X_1, X_2 correspond to the keys V, W sampled by $\mathcal{D}_L$, Y_1 corresponds to R and Z_1 corresponds to the message M.

It's worth noting that in the definition, nothing prevents us from writing an oracle that makes no sense mathematically. For example if oracle parameters Z_1, Z_2 correspond to handles for group elements, we can still return the product $Z_1 Z_2$ even though there is no suitable pairing that would allow the computation of this value, i.e. the value makes no sense. The tool doesn't care about this though, since it internally handles everything as formal Laurent polynomials over $\mathbb{Z}$, where such a product would have a meaning. In other words, the tool won't do any type checking of the return values of an oracle and assumes that the user writes down oracles that make sense.

Definition 33. An oracle is a tuple $\mathcal{O} = (q, n, m, r, l, \vec{F}, \vec{v}, \vec{w})$, where the different parts are defined as follows:

- q is the number of oracle queries allowed,
- n is the number of variables $A_1, \dots, A_n$ in $\mathbb{F}_p$ taken as a parameter,
- m is the number of values $Y_1, \dots, Y_m$ sampled in $\mathbb{F}_p$ by the oracle,
- r is the number of group handles $Z_1, \dots, Z_r$ taken as a parameter and the length of $\vec{w}$,
- l is the length of the vectors $\vec{F}$ and $\vec{v}$,
- $\vec{F}$ is a vector of Laurent polynomials in $\mathbb{F}_p[\vec{X}, \vec{X}^{-1}, \vec{A}, \vec{A}^{-1}, \vec{Y}, \vec{Y}^{-1}, \vec{Z}, \vec{Z}^{-1}]$.
- $\vec{v}$ is a vector of indices in I describing groups that oracle return values belong to,
- $\vec{w}$ is a vector of indices in I describing groups the Z_i belong to.

Here $\vec{v}$ contains the indices of the groups that each element of $\vec{F}$ returns a handle to and $\vec{w}$ contains the indices of the groups that each parameter Z_i contains a handle to.

The following describes the precise steps performed on each oracle query in the symbolic model:

1. On query $j \in [q]$ we create a new fresh variable $Y_{i,j}$ for each Y_i, where $i = 1, \dots, m$.
2. Denote by a_i the value of the parameter A_i, by z_i the value of the parameter Z_i and by x_i the value of each X_i sampled by $\mathcal{D}_L$.
3. We substitute x_i for each X_i, $Y_{i,j}$ for each Y_i, a_i for each A_i and $L_{w_i}[z_i]$ for each Z_i in the Laurent polynomials in $\vec{F}$.
4. We append the value of F_i after the substitution to L_{v_i}.

Since Theorem 24 captures generalized extraction problems and can be generalized to oracle queries, we can analyze such assumptions in the symbolic group model as before. As mentioned earlier, the symbolic version of the winning event can be obtained by plugging in the polynomials $L_{i_j}[h_j]$ for the variables Z_j instead of using the discrete logarithm.

Interactive Master Theorem.

To define the interactive master theorem, we introduce the notion of parametric completion. The *parametric completion* of $\mathbf{L}$ with respect to a group setting $\mathcal{GS}$ and an oracle $\mathcal{O}$ defined by $(q, n, m, r, l, \vec{F}, \vec{v}, \vec{w})$ is a family L_i of lists of Laurent polynomials defined as follows. Assume that in the oracle definition, $\vec{F}$ consists of Laurent polynomials in $\mathbb{F}_p[\vec{X}, \vec{X}^{-1}, \vec{A}, \vec{A}^{-1}, \vec{Y}, \vec{Y}^{-1}, \vec{Z}, \vec{Z}^{-1}]$. Define $\mathcal{A} = \{A_{i,j}\}$, where $i \in [n]$, $j \in [q]$, $\mathcal{Y} = \{Y_{i,j}\}$, $i \in [m]$, $j \in [q]$. The parametric completion then consists of lists of Laurent polynomials in

$$\mathbb{F}_p[\vec{X}, \vec{X}^{-1}, \mathcal{A}, \mathcal{A}^{-1}, \mathcal{Y}, \mathcal{Y}^{-1}, \mathcal{C}]$$

according to the algorithm described in Figure 2. The variables $\mathcal{C}$ are fresh variables created during the completion calculation, which represent the choice in the group elements that the adversary can give as a parameter to the oracle just like the variables in $\mathcal{A}$ represent the choice of constants given by the adversary as parameters. Their meaning becomes apparent in the following example.

Example 34. Assume that our group setting contains the single additive group $\mathbb{G}_1 = \mathbb{Z}/p\mathbb{Z}$ and an oracle that on input $x \in \mathbb{Z}/p\mathbb{Z}$ returns x^2. Such an oracle

```
$i = 0$
while $i < q$ :
    $\mathbf{L} = \mathcal{C}(\mathbf{L})$
    $s_j = |L_j| \; \forall j \in I$
    $Z'_j = \sum_{k=0}^{s_j - 1} C_{j,i,k} L_j[k]$, for fresh variables $C_{\cdot,\cdot,\cdot}$
    $F'_i = F_i[A_j \to A_{j,i}, Y_j \to Y_{j,i}, Z_j \to Z'_j]$
    Append $F'_j$ to $L_{w_j}$ $\forall j \in [l]$
    $i := i + 1$
$\mathbf{L} = \mathcal{C}(\mathbf{L})$
```

FIGURE 2: Parametric completion w.r.t. an oracle

would be described by the single polynomial $F_1(X_1, Z_1) = Z_1^2$. Assume that $L_1 = [1, X_1]$ upon initialization in the symbolic model. We have only one input list with no isomorphism or pairing, so the standard completion does nothing. On the first query, we have $s_1 = 2$, so we get

$$Z'_1 = C_{1,1,0}1 + C_{1,1,1}X_1$$

and

$$F'_1 = (C_{1,1,0}1 + C_{1,1,1}X_1)^2.$$

This implies that after one query $L_1 = [1, X_1, (C_{1,1,0}1 + C_{1,1,1}X_1)^2]$. Since the values of the $C_{\cdot,\cdot,\cdot}$ are chosen by the adversary, we see that anything computable by the adversary is a linear combination of the elements of L under some choices of the values of the $C_{\cdot,\cdot,\cdot}$. This can trivially be proven by induction. Given one more query, we would have

$$L_1 = [1, X_1, (C_{1,1,0}1 + C_{1,1,1}X_1)^2, (C_{1,2,0} + C_{1,2,1}X_1 + C_{1,2,2}(C_{1,1,0}1 + C_{1,1,1}X_1)^2)^2].$$

This also shows that given an oracle that takes group handle parameters, we can generally only handle problems under at most 3 oracle queries due to a typical combinatorial explotion in the number of constraints that our solver extracts from the problem. The number of constraints depends on the number of terms in the expanded expressions in L_1.

To state our interactive master theorem, we exploit that in the symbolic model, we can translate a generalized extraction problem to an equivalent generalized extraction problem where the adversary returns only elements in $\mathbb{F}_p$ and no handles. This is simply, because the setting is completely deterministic and because the adversary can return a handle to a polynomial element if and only if the polynomial is in the span of the completion. Thefore, instead of returning the handle to the polynomial, we can return the coefficients in the linear combination that constructs the polynomials from the completion and modify the winning condition accordingly. More precisely, let $\mathcal{C}^{\mathcal{O}}(\mathbf{L}) = \overline{L}_{i_1}, \ldots, \overline{L}_{i_l}$ denote the lists in the parametric completion. Assume now that the winning condition in the generalized extraction problem has a polynomial of the form

$$H(\vec{X}, \mathcal{Y}, \vec{B}, h_1, \ldots, h_n),$$

where h_j is a handle to some element in L_{i_j}. We replace H in the winning condition by

$$H'(\vec{X}, \mathcal{Y}, \vec{B}, \vec{\alpha}_1, \ldots, \vec{\alpha}_n) = H(\vec{X}, \mathcal{Y}, \vec{B}, \vec{\alpha}_1 \cdot \overline{L}_{i_1}, \ldots, \vec{\alpha}_n \cdot \overline{L}_{i_n}),$$

i.e. a handle to a polynomial is replaced by requiring the actual linear combination that computes it from the completion. Renaming variables, we can therefore assume in the symbolic case that the polynomials in the generalized extraction problem are of the form

$$H(\vec{X}, \mathcal{Y}, \vec{B}).$$

Finally, we note that an attack strategy in the symbolic model is given with respect to the sampled variables $\vec{X}$ and $\mathcal{Y}$ that are treated as formal variables. Any attack in the symbolic model requires a sequence of oracle queries, which is represented by instantiating $\mathcal{A}$ and $\mathcal{C}$ with some concrete values chosen by the adversary. We can encode the strategy directly in the polynomials in the generalized extraction problem by writing polynomials in the form

$$H(\vec{X}, \mathcal{Y}, \vec{B}, \mathcal{A}, \mathcal{C}),$$

i.e. we plug $\mathcal{A}$ and $\mathcal{C}$ into the parametric completion before plugging in for $\vec{X}$, $\mathcal{Y}$ and $\vec{B}$.

Theorem 35. *Let $\mathcal{GS}$ denote a group setting and let $\mathcal{D}_{\mathbf{L}}$ denote a polynomially-induced distribution. Consider the $(\hat{n}, \hat{m}, \vec{j}, \vec{H}, \vec{G})$-extraction problem in the generic and symbolic group models for $\mathcal{GS}$, $\mathcal{D}_{\mathbf{L}}$, and the oracle defined by $(q, n, m, r, l, \vec{F}, \vec{v}, \vec{w})$. Let $\vec{H}'$ and $\vec{G}'$ denote the translations of $\vec{H}$ and $\vec{G}$ with respect to this model that do not use handles. Then the problem is symbolically hard if there are no vectors $\vec{a}, \vec{b}, \vec{c}$ over $\mathbb{F}_p$ satisfying*

$$\left(\bigwedge_{j=1}^{|\vec{H}'|} H_j'(\vec{X}, \mathcal{Y}, \vec{b}, \vec{a}, \vec{c}) = 0\right) \wedge \left(\bigwedge_{j=1}^{|\vec{G}'|} G_j'(\vec{X}, \mathcal{Y}, \vec{b}, \vec{a}, \vec{c}) \neq 0\right).$$

In this case, the winning probability for the generic version is upper-bounded by

$$\frac{(s+q+q'l)^2 d}{2p} + \frac{(s+q'l)d_2}{p} + \frac{ed}{p},$$

where p is the group order, s is the sum of the sizes of the lists in $\mathbf{L}$, q the number of queries to the group-oracles, q' the number of queries to $\mathcal{O}$ and $e = |\vec{H}'| + |\vec{G}'|$. Finally, $d = d_1 + d_2$, where d_1 is an upper bound on the degree of the numerators and d_2 and upper bound on the degree of the denominators (in $\vec{X}$ and $\vec{Y}$) in any rational expression stored by the corresponding symbolic model and occuring in $\vec{H}'$ and $\vec{G}'$.

Proof. We use Theorem 24 to switch to the symbolic model which is equivalent up to the ϵ-bound stated in the theorem. Here, we take into account that each of the q' oracle calls adds l group elements to the lists and that the winning condition contains $e = |\vec{H}| + |\vec{G}|$ equality checks. Then, we show that if the side-condition condition is false, then there exists a symbolic adversary that wins with probability one. If the side-condition is true, then the winning probability in the symbolic model is always zero. To see why this is the case, first observe that an adversary that wins must query $\mathcal{O}$ with parameters corresponding to instantiating $\mathcal{A}$ and $\mathcal{C}$ with some $\vec{a}$ and $\vec{c}$ and return $\vec{b}$ such that the equalities $\vec{H}$ and the inequalities $\vec{G}$ are satisfied. Therefore, the side condition of the theorem exactly mirrors the winning condition. $\square$

3.5.1 *Automated Analysis*

To solve interactive problems, we use Gröbner basis techniques and SMT solvers to prove one of

1. No solution for all primes,
2. No solution for all primes except for some bad primes,
3. A solution over the rationals which can be converted into an attack for almost all primes, or
4. A solution over $\mathbb{C}$.

Even though we only encountered cases (1-3) in practice, case (4) is the reason for the incompleteness of our algorithm since the existence of a solution over $\mathbb{C}$ does not imply the existence of solutions over $\mathbb{F}_p$. Additionally, current Gröbner basis algorithms for a polynomial ring over $\mathbb{Z}$ are all exponential in the worst-case, but they tend to work reasonably well for ideals encountered in the wild. For this reason, in practice, our tool sometimes fails to give a response, in particular, when the number of queries is increased.

Our Method for Bounded Number of Queries.

We now describe the algorithm used by our tool to evaluate the side condition of Theorem 35.

1. We translate a given assumption to a group setting $\mathcal{GS}$, a list of polynomials $\mathbf{L}$, an oracle description $(q, n, m, r, l, \vec{F}, \vec{v}, \vec{w})$, and an extraction problem $(r, m, \{i_1, \ldots, i_m\}, \vec{H}, \vec{G})$.
2. We compute the parametric completion for a fixed number of queries q and translate the extraction polynomials $\vec{H} = (H_1, \ldots, H_k), \vec{G} = (G_1, \ldots, G_l)$ to obtain handle-free versions $\vec{H}' = (H_1', \ldots, H_k')$ and $\vec{G}' = (G_1', \ldots, G_l')$.
3. We extract the coefficients $\vec{f}_i, \vec{g}_j$ of the polynomials H_i', G_j' for $i = 1, \ldots, k$, $j = 1, \ldots, l$, represented as polynomials in $(\mathbb{Z}[\vec{A}, \vec{B}, \vec{C}])[\vec{X}, \vec{Y}]$. Writing $\vec{f}_i =$

$(f_{i,1}, \dots, f_{i,n_i})$ and $\vec{g}_j = (g_{j,1}, \dots, g_{j,m_j})$, where $f_{i,s}, g_{j,r} \in \mathbb{Z}[\vec{A}, \vec{B}, \vec{C}]$, we note that the side condition of Theorem 35 is satisfied iff there are no points $\vec{a}, \vec{b}, \vec{c}$ over $\mathbb{F}_p$ such that for all $i = 1, \dots, k$, $s \in \{1, \dots, n_i\}$, $f_{i,s}(\vec{a}, \vec{b}, \vec{c}) = 0$, and for all $j = 1, \dots, l$ there is some $s \in \{1, \dots, m_j\}$ such that $g_{j,s}(\vec{a}, \vec{b}, \vec{c}) \neq 0$.

4. We use the Rabinowitch trick to transfer inequalities to equalities by defining for each $g_{j,s}(\vec{A}, \vec{B}, \vec{C})$ the polynomial $\widetilde{g}_{j,s}(\vec{A}, \vec{B}, \vec{C})D_{j,s} - 1$ and set $\phi_j = \widetilde{g}_{j,1} \cdots \widetilde{g}_{j,m_j}$.

5. It is not hard to see that (over $\mathbb{F}_p$)

$$\begin{aligned} &\{(\vec{a}, \vec{b}, \vec{c}) \mid \bigwedge_{i,s} f_{i,s}(\vec{a}, \vec{b}, \vec{c}) = 0 \wedge \bigwedge_{j} \bigvee_{s} g_{j,s}(\vec{a}, \vec{b}, \vec{c}) \neq 0\} \\ = &\{(\vec{a}, \vec{b}, \vec{c}) \mid \exists \vec{d}. \bigwedge_{i,s} f_{i,s}(\vec{a}, \vec{b}, \vec{c}) = 0 \wedge \bigwedge_{j} \phi_j(\vec{a}, \vec{b}, \vec{c}, d_{j,1}, \dots, d_{j,m_j}) = 0\}. \end{aligned}$$

6. We compute a Gröbner basis I of the ideal generated by the $f_{i,s}$ and ϕ_j over $\mathbb{Z}$. We can then conclude as follows (for all adversaries performing up to q queries):
   a) If $I = (1)$, then return assumption is hard for all primes p.
   b) If I contains a constant $n \neq 1$, then return assumption is hard except for B consisting of all primes that divide n.
   c) If I does not contain a constant, we know that there is a solution over the complex numbers. We then try to find a solution over the rationals using a combination of primary decomposition of ideals, linear algebra, and model finding using SMT solvers. If we find a solution, we return the solution which describes an attack that works for almost all primes p. If we do not find a solution, then we return "unknown".

To summarize, our method is sound, i.e., whenever we return an attack or "hard except for B", then this is a valid conclusion. It is incomplete because the existence of a solution over the complex numbers does not imply the existence of a solution over $\mathbb{F}_p$. On the other hand, no solution over $\mathbb{C}$ implies no solutions over $\mathbb{F}_p$ when p is large enough. This is a consequence of the compactness theorem of first-order logic, i.e. Robinson's principle [Rob59].

3.6 COMPOSITE-ORDER GROUPS

We consider group settings generated by algorithms GroupGen that take a security parameter λ in unary representation and return a group setting $\mathcal{GS} = (N, \mathcal{G}, I, \Phi, \mathcal{E})$ where N is the product of distinct primes $p_1, \dots, p_k$, each greater than 2^λ, $\mathcal{G} = \{\mathbb{G}_i\}_{i \in \mathcal{I}}$ is a family of cyclic groups of composite-order N, Φ is a set of isomorphisms $\phi : \mathbb{G}_i \to \mathbb{G}_j$, and $\mathcal{E}$ is a set of admissible l-linear maps $e : \mathbb{G}_{i_1} \times \dots \times \mathbb{G}_{i_l} \to \mathbb{G}_{i_{l+1}}$. Additionally, GroupGen returns the prime factors $p_1, \dots, p_k$ of N as well as for each group $\mathbb{G}_i$ generators $P_{i,1}, \dots, P_{i,k}$ for the prime order subgroups. We assume that for a given group generator GroupGen, the index set $\mathcal{I}$, the number k of primes, and the types of isomorphisms and maps are fixed. We also assume that for each group $\mathbb{G}_i$, its description includes the generator $P_i = P_{i,1} + \dots + P_{i,k}$. Finally, we assume that the isomorphisms and maps are compatible with the generators, e.g., for $e : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_t$, it holds that $e(P_{i,r}, P_{j,r}) = P_{t,r}$.

An important property satisfied by bilinear pairings in our group settings is that $e(P_{i,r}, P_{j,s}) = 0$ for $r \neq s$. This is called the *canceling property* of composite-order pairings and follows from the fact that

$$p_r e(P_{i,r}, P_{j,s}) = e(p_r P_{i,r}, P_{j,s}) = e(0, P_{j,s}) = 0.$$

Analogously, $p_s e(P_{i,r}, P_{j,s}) = 0$, so choosing $a, b \in \mathbb{Z}$, s.t. $ap_r + bp_s = 1$, we get

$$e(P_{i,r}, P_{j,s}) = (ap_r + bp_s) e(P_{i,r}, P_{j,s}) = ap_r e(P_{i,r}, P_{j,s}) + bp_s e(P_{i,r}, P_{j,s}) = 0.$$

The implication of the canceling property is that computing a general pairing $e : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_t$ has the form

$$e(a_1 P_{i,1} + \dots + a_k P_{i,k}, b_1 P_{j,1} + \dots + b_k P_{j,k}) = a_1 b_1 P_{t,1} + \dots + a_k b_k P_{t,k}.$$

for $a_i, b_i \in \mathbb{Z}$, $i = 1, \dots, k$. Clearly, the same argument generalizes to multilinear pairings of higher arity. From the perspective of our tool, this means that we can process composite-order groups as lists of tuples and compute pairings by multiplying the tuples component by component.

3.6.1 *Generic and symbolic model for composite-order bilinear groups*

In the composite-order case, the generic group model is not defined for a concrete group setting, but the group setting is sampled on initialization. For a group generator GroupGen, a security parameter λ, and a distribution $\mathcal{D}$, we denote the corresponding generic group model with GGM GroupGen, $\lambda\mathcal{D}$. To define a symbolic version of this model, we restrict ourselves to polynomially induced distributions, which we define as follows for composite-order group settings.

Definition 36. Let $\mathcal{GS} = (N = p_1 \cdots p_k, \mathcal{G}, I, \Phi, \mathcal{E})$ be a composite-order group setting as defined above. Let $\mathbf{L} = \{L_i\}_{i \in \mathcal{I}}$ be a set of lists where each list element is a tuple of polynomials in $(\mathbb{F}_{p_1}[X_1, \ldots, X_n], \ldots, \mathbb{F}_{p_k}[X_1, \ldots, X_n])$. We define a distribution $\mathcal{D}_{\mathbf{L}}$ on $\mathcal{G} = \{\mathbb{G}_i\}_{i \in \mathcal{I}}$ as follows. Uniformly sample a point $\vec{x} \in \mathbb{Z}_N$, where $N = p_1 \cdots p_k$, and return $\mathbf{L}' = \{L'_i\}_{i \in \mathcal{I}}$, where $L'_i = [\Sigma_{j=1}^{k} f_j(\vec{x}) P_{i,j} \mid (f_1, \ldots, f_k) \leftarrow L_i]$. We say that a distribution $\mathcal{D}$ on $\{\mathbb{G}_i\}_{i \in \mathcal{I}}$ is *polynomially induced* if $\mathcal{D} = \mathcal{D}_{\mathbf{L}}$ for some $\mathbf{L}$.

We note that the definition of decisional, computational and generalized extraction problems directly translate to the product group setting.

Example 37. The Subgroup Decision Problem for 3 primes is defined as follows. A group generator returns $(\mathcal{GS}, \{p_1, p_2, p_3\}, \{g_{p_1}, g_{p_2}, g_{p_3}\}, \{h_{p_1}, h_{p_2}, h_{p_3}\}) \leftarrow$ GroupGen(λ), where the group setting is $\mathcal{GS} = (N = p_1p_2p_3, \{\mathbb{G}_1, \mathbb{G}_T\}, \{1, T\}, \emptyset, \{e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T\})$ and we assume that the description of $\mathbb{G}_1$ and $\mathbb{G}_T$ in the group setting includes generators $g = g_{p_1} + g_{p_2} + g_{p_3}$ and $h = h_{p_1} + h_{p_2} + h_{p_3}$. Our compatibility assumption, then translates to $e(g_{p_i}, g_{p_i}) = h_{p_i}$, $e(g_{p_i}, g_{p_j}) = 0$, $i \neq j$, and $e(g, g) = h$.

Using the additive notation in the definition above, the adversary is then given ag_{p_1}, bg_{p_3}, where $a, b \leftarrow \mathbb{Z}_N$, as well as $\mathcal{GS}$ and has to distinguish between $T_0 = cg_{p_1} + dg_{p_2}$ and $T_1 = cg_{p_1}$, where $c, d \leftarrow \mathbb{Z}_N$. We denote the distribution that samples T_i by $\mathcal{D}_i$ for $i = 0, 1$. Following our definition, the distribution $\mathcal{D}_0$ is described by the list of tuples of polynomials $L = [(1,1,1), (a,0,0), (0,0,b), (c,d,0)]$ and $L_T = [(1,1,1)]$ in $\mathbb{Z}[a,b,c,d]^3$, where the tuples $(1,1,1)$ corresponds to the generators g and h given as part of the group description. $\mathcal{D}_1$ is described similarly, except that $L = [(1,1,1), (a,0,0), (0,0,b), (c,0,0)]$.

From Generic to Symbolic Group Models

We define the symbolic group model $\mathrm{Sym}_{\mathcal{D}_\mathbf{L}}^{\mathrm{GroupGen},\lambda}$ for a group generator GroupGen, a security parameter λ and a polynomially induced distribution $\mathcal{D}_\mathbf{L}$ as follows.

- Sample primes $p_1, \dots, p_k$ according to the same distribution as GroupGen.
- Internally store lists of vectors of polynomials in $\mathbb{Z}_N[X_1, \dots, X_n]^k$ where $X_1, \dots, X_n$ are the variables occurring in $\mathbf{L}$ and $N = p_1 \cdots p_k$.
- The oracles perform addition, negation, and equality checks in the product ring of the polynomial rings.
- The oracle $\mathrm{isom}_\phi(h)$ copies elements from one list to another.
- The oracle $\mathrm{map}_e(h_1, \dots, h_k)$ appends the product of the source list elements to the target list.

Example 38. The symbolic version of the Subgroup Decision Problem for 3 primes in Example 37 is defined by letting the internal state of the oracle be initialized to $L = [(1,1,1),(a,0,0),(0,0,b),(c,d,0)]$ and $L_T = [(1,1,1)]$ in $\mathbb{Z}_N[a,b,c,d]^3$, where N is the order of the groups in the corresponding group setting.

We now prove that it is hard to distinguish the generic group model and the corresponding symbolic group model under the assumption that it is hard to factor the composite numbers N sampled by GroupGen.

Theorem 39. *Let λ be a security parameter and let $\mathcal{GS} = (N, \mathcal{G}, I, \Phi, \mathcal{E})$ denote a group setting obtained by running* $\mathrm{GroupGen}(\lambda)$*, where each $\mathbb{G} \in \mathcal{G}$ is cyclic of degree $N = p_1 \cdots p_k$, where the p_i, $i = 1, \dots, k$ are distinct primes of size λ. Let $\mathcal{D}_\mathbf{L}$ be a polynomially induced distribution, $s = \sum_{i \in \mathcal{I}} |L_i|$ and q the total number of queries performed by an adversary $\mathcal{A}$, then*

$$\left| \Pr[\mathrm{Sym}_{\mathcal{D}_\mathbf{L}}^{\mathrm{GroupGen},\lambda}(\mathcal{A}) = 1] - \Pr[\mathrm{Gen}_{\mathcal{D}_\mathbf{L}}^{\mathrm{GroupGen},\lambda}(\mathcal{A}) = 1] \right| \leqslant \frac{(s+q)^2 d^k}{2 p_1 \cdots p_k} + \varepsilon(\lambda),$$

where $\varepsilon(\lambda)$ is an upper bound on the probability that $\mathcal{A}$ succeeds in factoring N.

Proof. Translating composite-order assumptions in a group setting $\mathcal{GS} = (N, \mathcal{G}, I, \Phi, \mathcal{E})$, where the initial state of the generic group is sampled by a polynomially induced distribution $\mathcal{D}_L$, to a symbolic model is based on the following sequence of games.

Game 0: Real game.

Game 1: Replace the internal representations of the groups by the additive group $\mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_k}$.

Game 2: Replace all sampled elements by formal variables and sample values for them in $\mathbb{Z}_N$. Replace the internal representation of the groups by $(\mathbb{Z}_{p_1}[\overline{X}], \dots, \mathbb{Z}_{p_k}[\overline{X}])$, where $\overline{X}$ denotes the set of formal variables. Perform equality check on polynomial tuples evaluated at the sampled values.

Game 3: Replace the equality check oracle in the previous game to perform an equality check on the tuples of formal polynomials.

Game 4: Replace the polynomial computations to happen over $(\mathbb{Z}_N[\overline{X}], \dots, \mathbb{Z}_N[\overline{X}])$

We note that games 0–2 are indistinguishable. Let d be the highest degree of any polynomial appearing as a component of a k-tuple. If we plug in randomly sampled values for the variables, then the probability that component i is zero is by Schwarz-Zippel bounded above by d/p_i and the probability that all components are zero is $d^k/(p_1 \cdots p_k)$, since by the Chinese Remainder Theorem, the value of each variable X modulo p_i is independent of the value modulo p_j for $i \neq j$. Let q be the number of queries and $s = \sum_{i \in \mathcal{I}} |L_i|$ the total initial lengths of the internal lists. The number of polynomial tuples in the problem is then limited by $s + q$, so the distinguishing probability of Game 2 and 3 is bounded by

$$\frac{(s+q)^2 d^k}{2 p_1 \cdots p_k}.$$

Note that any adversary that can distinguish between Game 3 and 4 can be used to efficiently factor N, which gives us the second term in the bound. □

3.6.2 *Master Theorem*

Let $\mathcal{D}_{\mathbf{L}}$ and $\mathcal{D}_{\mathbf{L}'}$ be two polynomially induced distributions such that $|L_i| = |L'_i|$ for all i. In what follows we derive a condition under which

$$|\Pr[\text{Game } 4^{\text{GroupGen},\lambda}_{\mathcal{D}_{\mathbf{L}}}(\mathcal{A}) = 1] - \Pr[\text{Game } 4^{\text{GroupGen},\lambda}_{\mathcal{D}_{\mathbf{L}'}}(\mathcal{A}) = 1]| = 0$$

with Game 4 defined as in the proof of Theorem 39.

In the setting of Game 4 computing pairings corresponds to component-wise multiplication of tuples. This allows us to directly extend our concept of completion to tuples. From example 25 we directly see that computing the completion for tuples can be done in exactly the same way, except that we plug in tuple values for the freshly chosen variables. In the composite-order symbolic model the completion on the product group structure similarly records the possible elements that the adversary may compute from the inputs. It follows that the left and right version of Game 4 are equally distributed if

$$\{\vec{\alpha} \in \mathbb{Z}_N^r \mid \vec{\alpha} \cdot \mathcal{C}(\mathbf{L})_t\} = \{\vec{\alpha} \in \mathbb{Z}_N^r \mid \vec{\alpha} \cdot \mathcal{C}(\mathbf{L}')_t\},$$

where $r = |\mathcal{C}(\mathbf{L})_t| = |\mathcal{C}(\mathbf{L}')_t|$.

Finally, by combining Theorem 39 and the above argument, we obtain the following version of the master theorem for the composite-order setting:

Theorem 40. *Let λ be a security parameter and let $\mathcal{GS} = (N, \mathcal{G}, I, \Phi, \mathcal{E})$ denote a group setting obtained by running* GroupGen(λ)*, where each $\mathbb{G} \in \mathcal{G}$ is cyclic of degree $N = p_1 \cdots p_k$, where the p_i, $i = 1, \ldots, k$ are distinct primes of size λ. Let $\mathcal{D}_{\mathbf{L}}, \mathcal{D}_{\mathbf{L}'}$ be polynomially-induced distributions such that $|L_i| = |L'_i|$ for all $i \in \mathcal{I}$. Let t denote the index of the target group, $s = \sum_{i\in\mathcal{I}} |L_i|$, $r = |\mathcal{C}(\mathbf{L})_t|$, and let d denote an upper bound for the total degrees of the polynomials in the completions of the lists. If*

$$\{\vec{a} \in \mathbb{Z}_N^r \mid \vec{a} \cdot \mathcal{C}(\mathbf{L})_t = 0\} = \{\vec{a} \in \mathbb{Z}_N^r \mid \vec{a} \cdot \mathcal{C}(\mathbf{L}')_t = 0\},$$

then

$$|\Pr[\mathrm{Gen}^{\mathrm{GroupGen},\lambda}_{\mathcal{D}_L}(\mathcal{A}) = 1] - \Pr[\mathrm{Gen}^{\mathrm{GroupGen},\lambda}_{\mathcal{D}_{L'}}(\mathcal{A}) = 1]| \leqslant \frac{(s+q)^2 d^k}{p_1 \cdots p_k} + 2\varepsilon(\lambda).$$

for all adversaries $\mathcal{A}$ that perform at most q operations and with probability $\varepsilon(\lambda)$ of factoring N.

3.6.3 *Non-parametric and Non-interactive Assumptions*

The algorithm in section 3.3 extends directly with one simple modification. For the polynomials in the closure a basis is given by all the monomials appearing in the closure. We just need to extend this to tuples, i.e. for any element $(P_1, \dots, P_k)$ in the closure and any monomial M appearing in P_i, we add $(0, \dots, 0, M, 0, \dots, 0)$ to the basis, where M is in the ith position. In the resulting basis, we can express all the polynomial tuples in the closure and the algorithm extends with no further modifications.

3.7 RATIONAL INPUT DISTRIBUTIONS

Assume that we work in a (leveled) k-linear model. We can translate an assumption with rational exponents to a symbolic model with polynomial exponents. If we treat the sampled field elements as variables, we can compute an expression Q with the property that any input at level i is cleared of its denominator when multiplied by Q^i. Since we can clear denominators with Q, we can rewrite all inputs with Q^i in the denominator. This does not change the inputs, since they are still precisely the same elements in the field. Note also that Q is only zero if one of the denominators are zero.

Example 41. Given $1/x$ at level 1 and $1/x^2$ at level 3, we choose $Q = x$. We may rewrite the elements as $1/Q$ and x/Q^3.

We can now apply the following sequence of games to reduce to the symbolic model.

GAME 0: The original game defined by the assumption.

GAME 1: Rewrite the elements to have Q^i in the denominator at level i. This is still exactly the same game, since the input ele

GAME 2: Replace the sampled field elements by variables. Do equality checks by plugging in their sampled values.

GAME 3: Switch to symbolic model using Schwartz-Zippel.

GAME 4: Throw away all denominators in the inputs in Game 3.

Games 0–1 are indistinguishable and so are Game 1–2 unless Q is zero at the sampled points. Game 4 is in the polynomial symbolic group model and we can continue as before. The crucial step is from Game 3 to Game 4. These two games are indistinguishable, since we can only compare things that are at the same level. By construction (similarly for a k-linear pairing)

$$e(a/Q^i, b/Q^j) = ab/Q^{i+j}$$

so elements at each level that we can compute always have the "correct" denominators. Since denominators are always equal at each level, they will have no impact on equality queries and may just be dropped. We only need to be a bit careful about the Schwartz-Zippel step. We will be comparing elements of the form $P/Q^i = R/Q^i$. We only fail to simulate correctly if either Q is zero at the sampled points or $P = R$ with $P \neq R$ as polynomials. We see that it suffices to compute the highest degree d that we can get in the numerator in Game 2, add $\deg Q$ to it, and use this for the degree in Schwartz-Zippel.

Finally, we note that a general group setting ignoring Φ can be expressed as a graph with nodes $\{\mathbb{G}_i\}$ and edges from $\mathbb{G}_{i_j}$ to $\mathbb{G}_{i_{n+1}}$, $j = 1, \ldots, n$, if $\exists e : \mathbb{G}_{i_1} \times \cdots \times \mathbb{G}_{i_n} \to \mathbb{G}_{i_{n+1}} \in \mathcal{E}$. In this case, we must assume that the resulting graph is directed and acyclic and has the property that any two paths between two nodes must have the same length. The latter property is required in order to assign a "level" for each group.

Assumption	Source	Type	Result	Runtime
DBDH	[BF01]	NP	valid	2.3s
Freeman 3	[Fre10]	NP	valid ($p > 2$)	89.2s
Freeman 4	[Fre10]	NP	valid ($p > 2$)	13.9s
k-lin, $k \in \{1, 2, 3, 4\}$	[Sha07]	NP	valid	67.1s
2-lin w/ 3-lin map		NP	attack	3.3s
2-SCasc	[EHK+13]	NP	valid	2.3s
2-BDH	[BSW13]	NP	valid	2.8s
l-DHE	[BB04]	P	valid	0.2s
l-DHI	[BGW05]	P	valid	0.1s
(l, k)-MMDHE	[HSW13]	P	valid	3.7s
5-SDH	[BB04]	GE	valid	2.3s
LRSW	[LRSW99]	I	valid ($q = 4$)	2.3s
m-LRSW	[BGOY07a]	I	attack ($q = 2$)	0.1s
IBSAS	[BGOY07b]	I	valid ($q = 3$)	28.8s
Strong-LRSW	[ACdM05]	I	valid ($q = 4$)	2.8s
s-LRSW	[GT12]	I	valid ($q = 4$)	2.3s
KSW, Assumption 1	[KSW13]	C	valid	4.5s

TABLE 1: Analysis for various assumptions from the literature. In the Type column, NP = nonparametric, P = parametric, GE = generalized extraction, I = interactive, C = composite.

3.8 EXAMPLES

The purpose of this section is to show how the tool can handle real-world cryptographic assumptions. Table 1 shows a fairly versatile set of assumptions extracted from the literature together with the runtime of the tool. For interactive assumptions we note that the runtime is strongly dependent on the number of oracle queries allowed to be performed by the attacker with significant slowdown typically starting when $q = 3$ or $q = 4$.

Example 42. We start by describing the 2-SCasc assumption of [EHK+13], since it shows how an assumption given in a non-standard format can be translated to a form the Generic Group Analyzer can understand. Let $\mathbb{G}$ be some group of order p, g a generator, and write

$$A = \begin{bmatrix} a & 0 \\ 1 & a \\ 0 & 1 \end{bmatrix}.$$

For a matrix $B = (b_{i,j})$ with $b_{i,j} \in \mathbb{Z}_p$, we write $[B]$ for the matrix $(g^{b_{i,j}})$. We then say that the 2-SCasc assumption holds in $\mathbb{G}$ if it's infeasible for an adversary to distinguish $[\vec{r}]$ from $[\vec{u}]$ given $[A]$, where

$$\vec{r} = A = \begin{bmatrix} a & 0 \\ 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ x + ay \\ y \end{bmatrix}, \quad \vec{u} = \begin{bmatrix} x \\ y \\ z \end{bmatrix},$$

where $a, x, y, z \leftarrow \mathbb{Z}_p$. In traditional notation, we can describe the assumption as the following two lists of tuples being computationally indistinguishable

$$(g, g^a, g^{ax}, g^{x+ay}, g^y), \; (g, g^a, g^x, g^y, g^z),$$

where $a, x, y, z \leftarrow \mathbb{Z}_p$.

To describe the assumption in the input language of the Generic Group Analyzer, in a setting, where we assume that $\mathbb{G}$ is equipped with a pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, we would have the following:

```
maps G * G -> GT.


input [ a ] in G.
input_left [ a*x, x + a*y, y ] in G.
input_right [ x, y, z ] in G.
```

Note that we do not need to include the 1 in the common input in $\mathbb{G}$, since the tool implicitly assumes that the generator has been given.

Example 43. A simple parametric assumption is l-DHE [BB04], which is described as follows. We say that l-DHE is hard in a bilinear group $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ of order

p with generators $g, h \in \mathbb{G}$, if given g^{x^i}, $i = 1, \ldots, l-1, l+1, \ldots, 2l$, it's hard to distinguish $e(g,h)^{x^l}$ from random, where $x \leftarrow \mathbb{Z}_p$. If we write $h = g^y$, then the assumption can be translated into

```
setting symmetric.
levels 2.
problem_type decisional.

input
  [ 1
  , Y
  , forall i in [0, l - 1]: X^i
  , forall j in [l + 1, 2*l]: X^j ] @ 1.

challenge Y*X^l @ 2.
```

In the snippet of code above, we simply model a bilinear group as a two-leveled multilinear group.

Example 44. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear group of order $n = pqr$, where p, q, r are prime. Let g_p, g_q, g_r be generators of the p, q, r order subgroups of $\mathbb{G}$. Then, Assumption 1 in [KSW13], is hard we choose random elements $Q_1, Q_2, Q_3 \in \mathbb{G}_q$, $R_1, R_2, R_3 \in \mathbb{G}_r$, and give n out to an adversary together with

$$g_p, g_r, g_q R_1, g_p^b, g_p^{b^2}, g_p^a, g_q, g_p^{ab} Q_1, g_p^s, g_p^{bs} Q_2 R_2,$$

then it's hard for the adversary distinguish $T = G_p^{b^2 s} R_3$ from $T' = G_p^{b^2 s} Q_3, R_3$, where $a, b, c \leftarrow \mathbb{Z}_p^s$. We note that the Generic Group Analyzer choses a canonical set of generators for the p, q, r order subgroups that are given to the adversary. If we denote by x, y, z the discrete logarithms of g_p, g_q, g_r with respect to the canonical generators, then we can express the assumption as follows:

```
map G1 * G1 -> GT.
composite 3.
```

```
input [ (X,0,0), (0,0,Z), (0,Y,Z*R1), (X*b,0,0), (X*b^2,0,0), (X*a,Y,0),
        (X*a*b,Y*Q1,0), (X*s,0,0), (X*b*s, Y*Q2, Z*R2) ] in G1.

input_left [ (X*b^2*s, 0, Z*R3) ] in G1.
input_right [ (X*b^2*s, Y*Q3, Z*R3) ] in G1.
```

Examples of how to describe interactive assumptions with complicated winning conditions can be found in Section 4.5.

4 STRUCTURE-PRESERVING SIGNATURE SCHEMES

Structure-preserving signatures [AFG+10] (SPS) are signature schemes defined over bilinear groups in which messages, public keys and signatures are all group elements, and the verification algorithm consists of evaluating pairing-product equations. One of the main motivations of considering such specific signature schemes is that they are remarkably useful in the modular design of several cryptographic protocols, notably in combination with non-interactive zero-knowledge (NIZK) proofs of knowledge about group elements, and more specifically with the Groth-Sahai proof system.

Realization of SPS has been considered over the three possible bilinear groups settings introduced in the classification of Galbraith, Paterson and Smart [GPS08]. Recent work has focused on proving lower bounds on the complexity of SPS, and exhibiting optimal constructions that match lower bounds. The common measures of complexity adopted in all these works are the number of group elements in the public key, the number of group elements in the signature, and the number of pairing-product equations in the verification algorithm.

This chapter is based on the results in [BFF+15]. Instead of analyzing the number of pairing-product equations required, we analyze the number of pairings that need to be performed by the verifier during signature verification. As previous work only considers the number of verification equations, this does not shed much light on the number of pairings required, since a verification equation might be very complicated and require many pairings for verification. Intuitively though, one would expect fewer equations to lead the fewer pairings. To close this gap in the analysis, we consider the Type II case, specifically, we look at signatures in the Type II setting of optimal bandwidth. For such signatures we prove lower bounds and prove tightness of these bounds by providing signatures secure in the generic group model, which have verification equations matching the lower bounds. A key part of this work is using automation provided by our generic group analyzer tool. Specifically, the conjectured lower bound is a consequence of trying to synthesize schemes with low number of pairings. Additionally, our new EUF-CMA secure scheme matching the proven lower bound is found through exhaustive search.

4.1 PRELIMINARIES

Given a bilinear group $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, a structure-preserving signature scheme is a signature scheme, where the verification key, the messages and the signatures consist only of group elements from $\mathbb{G}_1$ and $\mathbb{G}_2$. The verification algorithm evaluates the signature by deciding group membership of elements in the signature, and by evaluating pairing product equations, which are equations of the form:

$$\prod_i \prod_j e(X_i, Y_j)^{a_{ij}} = 1,$$

where $X_1, X_2, \ldots \in \mathbb{G}_1$, $Y_1, Y_2, \ldots \in \mathbb{G}_2$ are group elements appearing in *PP*, *VK*, M and σ. Note that in the Type II setting it may hold that $X_i = \phi(Y_j)$ for some i, j. Furthermore, we assume that the elements $a_{ij} \in \mathbb{Z}_p$ are constants stored in *PP*. More precisely:

Definition 45 (Structure-preserving signatures (SPS))**.** A signature scheme (Setup, KeyGen, Sign, Verify) is said to be *structure-preserving* with respect to some bilinear group generator $\mathcal{G}$ if

- *PP*: Consists of a bilinear group $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H)$ generated by $\mathcal{G}$ and a set of constants in $\mathbb{Z}_p$,

- *VK*: The verification key consists of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$,

- $\mathcal{M}$: Messages consist of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$,

- σ: Signatures consist of group elements in $\mathbb{G}_1$ and $\mathbb{G}_2$,

- Sign : The signing algorithm only uses generic group operations.[1]
- Verify : The verification algorithm only needs to decide membership in $\mathbb{G}_1$ and $\mathbb{G}_2$, use the homomorphism ψ (in the Type II setting), and evaluate pairing product equations.

4.2 KNOWN LOWER BOUNDS OF TYPE II SPS

A number of lower bounds on signature size, verification key size and the number of verification equations have been established for secure structure-preserving signature schemes and one-time signature schemes. In particular, Abe et al. [AGOT14a] establish many such bounds in the Type II setting. As some of our results rely heavily on those bounds, we recall them below. Note that membership tests are not counted as "verification equations", although some of them may require an amortizable pairing computation in practical instantiations.

First, just as Type I and Type III SPS, Type II SPS for messages in $\mathbb{G}_1$ require two verification equations:

Lemma 46 ([AGOT14a, Theorem 3])**.** *A structure-preserving signature scheme for messages in $\mathbb{G}_1$ must have at least two verification equations. This holds even for one-time signatures with security against random message attack.*

Since we will focus on schemes with a single verification equation, we will therefore consider signatures on messages in $\mathbb{G}_2$, which can have a single verification equation. In that case, Abe et al. obtain a lower bound on verification key size, and show that all signature elements must be in $\mathbb{G}_2$.

Lemma 47 ([AGOT14a, Theorem 4 and Lemma 1])**.** *A structure-preserving signature scheme with a single verification equation must have at least two group elements in the verification key, and can have no non-redundant signature element in $\mathbb{G}_1$. This holds even for one-time signatures secure under random message attack.*

1 Technically, this condition was not required in the original definition of Abe et al. [AFG$^+$10], but all known constructions satisfy this property and it is required for the proofs of most lower bounds to go through. Since an SPS scheme with a non-generic signing algorithm would be very unnatural and surprising, it seems appropriate to include genericity of the signer in the definition (see also the discussion in [AGOT14a, §2.3]).

Finally, signatures in a secure Type II SPS scheme must consist of at least two group.

Lemma 48 ([AGOT14a, Theorem 5]). *An* EUF-RMA-*secure structure-preserving signature scheme must have at least* 2 *group elements for messages in* $\mathbb{G}_2$ *and at least* 3 *group elements for messages in* $\mathbb{G}_1$*.*

4.3 LOWER BOUNDS ON THE NUMBER OF PAIRINGS

We now show lower bounds for the number of pairings in the pairing-product verification equations of SPS in the Type II setting. In particular, in our analysis we consider SPS schemes that already match the lower bounds shown in [AGOT14a], i.e., they have 2 group elements in the verification key, 2 group elements in the signature and the verification consists of a single pairing-product equation (as well as possible group membership tests).

Definition 49. A pairing in a verification equation of an SPS scheme is *offline* if if its parameters only consists of group elements in PP as well as VK. Any pairing not satisfying this requirement is called *online*.

The previous classification simply takes into account that certain pairings can be computed once if we verify for example many messages signed using the same public parameters and signing key. However, if a pairing in a verification equation takes as a parameter either the message to be verified or its signature, then the parameters will be different for each message, regardless of whether they are signed with the same key and public parameters. Therefore, we can't avoid computing these pairings by e.g. caching pairings computed in earlier signature verifications.

4.3.1 *Main result*

Having defined the notion of online and offline pairings, we are now ready to state our main result. It shows that any optimal-size SPS scheme requires at least three pairings for verification, and two of these pairings must be online ones.

Theorem 50 (Main result)**.** *Any* EUF-CMA*-secure structure-preserving signature scheme in the Type II setting with* 1 *verification pairing-product equation,* 2 *group elements in the verification key and* 2 *elements in the signature requires at least* 3 *pairings in the pairing-product equation, and at least* 2 *of them must be online pairings.*

To prove the theorem, we distinguish between three cases according to which groups the two elements V, W of the verification key belong to, i.e., (i) $V, W \in \mathbb{G}_2$, (ii) $V, W \in \mathbb{G}_1$, and (iii) $V \in \mathbb{G}_1, W \in \mathbb{G}_2$.

The first case is rather simple and is addressed in the following lemma which shows that there exists no such structure-preserving signature scheme.

Lemma 51. *There is no secure structure-preserving signature scheme in the Type II setting with a single verification equation and a verification key consisting entirely of elements of* $\mathbb{G}_2$*.*

Proof. We know by Lemma 47 and Lemma 48 that the signatures and messages all have to be in $\mathbb{G}_2$. Since all inputs are in $\mathbb{G}_2$ the scheme would also be secure in the Type I setting and must therefore be insecure since Type I SPS require two pairing product equations for security [AGOT14b, Theorem 4]. □

The second case is somewhat more involved, and mainly addressed by the following lemma, proved in Section 4.3.3 below.

Lemma 52. *An* EUF-CMA*-secure structure-preserving signature scheme in the Type II setting with* 1 *verification pairing-product equation,* 2 *group elements* $V, W \in \mathbb{G}_1$ *in the verification key and* 2 *group elements in the signature requires at least* 3 *pairings in the pairing-product equation.*

The previous lemma establishes that 3 pairings are needed, so all that remains to show is that 2 of them must be online. This follows immediately from the following observation.

Lemma 53. *In a Type II structure-preserving signature scheme where all verification key elements are in* $\mathbb{G}_1$*, it is possible to compute all the offline pairings in a verification pairing-product equation using a single pairing evaluation. More generally,* $\ell + 1$ *pairing evaluations are sufficient if the verification key contains* ℓ *elements in* $\mathbb{G}_2$*.*

Proof. Indeed, if the verification key is $(V_1, \ldots, V_k, U_1, \ldots, U_\ell) \in \mathbb{G}_1^k \times \mathbb{G}_2^\ell$, then any product of offline pairings can be expanded into an expression of the form $\prod_{i,j} e(X_i, Y_j)^{c_{ij}}$ where Y_j runs through $H, U_1, \ldots, U_\ell$ (since these are the only elements of $\mathbb{G}_2$ in the verification key and the public parameters) and X_i runs through $G, V_1, \ldots, V_k, \psi(U_1), \ldots, \psi(U_\ell)$. By rewriting the product as:

$$\prod_j e\Big(\prod_i X_i^{c_{ij}}, Y_j\Big)$$

we can compute it with at most $\ell + 1$ pairing evaluations as required. $\square$

Finally, to complete the proof of Theorem 50, we only need to prove it when the verification key consists of one element of $\mathbb{G}_1$ and one element of $\mathbb{G}_2$. This case, which is somewhat less interesting in practice as such a scheme is less space efficient than when all key elements are in $\mathbb{G}_1$, but turns out to be more technically challenging, is dealt with in detail in Section 4.4.

4.3.2 *Gaps in Bounds Between* EUF-RMA *and* EUF-CMA-*Security*

It is worth noting that Lemma 52, in the setting when $(V, W) \in \mathbb{G}_1^2$, holds only for EUF-CMA-secure SPS schemes, which forces EUF-CMA-security into the statement of Theorem 50. However, for the setting $(V, W) \in \mathbb{G}_1 \times \mathbb{G}_2$, as discussed in Section 4.4, the lower bound holds even for EUF-RMA-secure SPS. In this section we therefore investigate the case of EUF-RMA-security for the $(V, W) \in \mathbb{G}_1^2$ setting. We prove next a slightly weaker lower bound in this setting for the case of EUF-RMA-secure SPS.

Theorem 54. *Any* EUF-RMA-*secure structure-preserving signature scheme in the Type II setting with* 1 *verification pairing-product equation requires at least* 2 *pairings in the pairing-product equation, and both of them must be online pairings.*

Proof. It suffices to show that a Type II SPS scheme with a single verification equation (and which we can assume without loss of generality signs one-element messages) cannot be EUF-RMA-secure if the pairing-product equation consists of only one online pairing (and any number of offline pairings).

To see this, denote by $(S_1, \dots, S_k)$ the signature vector (which is in $\mathbb{G}_2^k$ without loss of generality by Lemma 47), and observe that the pairing product equation must be of the form:

$$\prod_{i,j} e(X_i, Y_j) = e\Big(\psi(M)^{a_0} \cdot \prod_{i=1}^{k} \psi(S_i)^{a_i} \cdot Z, M^{b_0} \cdot \prod_{j=1}^{k} \psi(S_j)^{b_j} \cdot T\Big)$$

where the pairings on the left-hand side are offline (and hence the X_i's and Y_j's do not depend on the message or the signature), and Z, T are elements which also do not depend on the message or the signature. But then we can do the change of variables:

$$(R', S') = \Big(\psi(M)^{a_0} \cdot \prod_{i=1}^{k} \psi(S_i)^{a_i}, M^{b_0} \cdot \prod_{j=1}^{k} \psi(S_j)^{b_j}\Big)$$

and then (R', S') provides a two-element EUF-RMA-secure signature scheme whose verification equation is just:

$$\prod_{i,j} e(X_i, Y_j) = e(R' \cdot Z, S' \cdot T),$$

and in particular does not depend on the message: this is a contradiction. $\square$

We see that the bounds given by Theorem 50 and Theorem 54 show a gap. Namely, there could exist an EUF-RMA-secure scheme with precisely two online pairings and no offline pairing. We confirm that both lower bounds are indeed tight, by providing an EUF-RMA-secure SPS with precisely two online pairings in Section 4.5.4.

4.3.3 *Proof of Lemma 52*

Proof. The proof proceeds by contradiction showing that having a scheme with only 2 pairings in the single pairing-product equation is impossible. Let us first recall that in this setting we have a message $M \in \mathbb{G}_2$, verification keys $V, W \in \mathbb{G}_1$ and signature elements $R, S \in \mathbb{G}_2$. As usual, we denote their discrete logarithms by the corresponding lower case letters. We may write the general verification equation in terms of the discrete logarithms of M, R, S, V, W as follows:

$$\begin{aligned}(c_1 m + c_2 r + c_3 s + c_4 v + v_5 w + c_6)(d_1 m + d_2 r + d_3 s + d_4) = \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_4),\end{aligned} \tag{2}$$

where the products represent a pairing, the left factor in each product represent the element in $\mathbb{G}_1$ and the right factor the element in $\mathbb{G}_2$ of each pairing.

Now if we define the vectors $X_1 = (c_1, \dots, c_6), X_2 = (e_1, \dots, e_6), Y_1 = (d_1, \dots, d_4), Y_2 = (f_1, \dots, f_4)$ over $\mathbb{Z}_p$ and the matrix:

$$E = X_1^t Y_1 - X_2^t Y_2,$$

then the verification equation (2) can be rewritten as

$$(m, r, s, v, w, 1)^t \cdot E \cdot (m, r, s, 1) = 0.$$

A simple observation shows that if $\operatorname{Ker} E$ contains a vector $(x_1, \dots, x_4)$, where $x_4 \neq 0$, then we may scale to $x_4 = 1$ and then

$$m = x_1, \ r = x_2, \ s = x_3$$

is a valid key-only attack forgery (since the kernel of E can be computed entirely from the public parameters). It follows that if the scheme is secure, then $\operatorname{Ker} E \subset \{(x_1, \dots, x_4) \mid x_4 = 0\}$. However, this implies that the following system

$$\begin{cases} d_1 m + d_2 r + d_3 s = -d_4 \\ f_1 m + f_2 r + f_3 s = -f_4 \end{cases}$$

lacks a solution, since otherwise

$$E(m, r, s, 1) = X_1^t Y_1 (m, r, s, 1) - X_2^t Y_2 (m, r, s, 1) = 0 - 0 = 0.$$

Up to exchanging the roles of Y_1 and Y_2 without loss of generality, this implies that $Y_1 = cY_2 + (0,0,0,\lambda)$ for some constants c, λ, and hence:

$$E = X_1^t Y_1 - X_2^t Y_2 = X_1^t\big(cY_2 + (0,0,0,\lambda)\big) - X_2^t Y_2 = (\lambda X_1)^t \cdot (0,0,0,1) - (X_2 - cX_1)^t Y_2.$$

Therefore, after relabeling the coefficients, we may assume that $Y_1 = (0,0,0,1)$ and the verification equation (2) can be rewritten as

$$\begin{gathered} c_1 m + c_2 r + c_3 s + c_4 v + v_5 w + c_6 = \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_4), \end{gathered} \tag{3}$$

Now if $(c_4, c_5) = \lambda(e_4, e_5)$ or $(e_4, e_5) = \lambda(c_4, c_5)$, then we may replace the verification key by a single element $t = e_4 v + e_5 w$ or $t = c_4 v + c_5 w$. For example, if $(c_4, c_5) = \lambda(e_4, e_5)$, set $t = e_4 v + e_5 w$ and write the verification as

$$\begin{gathered} c_1 m + c_2 r + c_3 s + \lambda t + c_6 = \\ (e_1 m + e_2 r + e_3 s + t + e_6)(f_1 m + f_2 r + f_3 s + f_4), \end{gathered}$$

which is insecure by Lemma 47. It follows that

$$\det \begin{pmatrix} c_4 & c_5 \\ e_4 & e_5 \end{pmatrix} \neq 0$$

and we may do a linear change of variables

$$\begin{pmatrix} v' \\ w' \end{pmatrix} = \begin{pmatrix} c_4 & c_5 \\ e_4 & e_5 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} + \begin{pmatrix} c_6 \\ e_6 \end{pmatrix},$$

so that the verification equation (3) becomes, after renaming coefficients,

$$c_1 m + c_2 r + c_3 s + v = (e_1 m + e_2 r + e_3 s + w)(f_1 m + f_2 r + f_3 s + f_4). \tag{4}$$

Note that the vectors $(c_2, c_3), (e_2, e_3), (f_2, f_3)$ cannot all be collinear, because otherwise we may again compress the signature into one group element as above, which we already know is impossible.

Next, we look at the matrix

$$N = \begin{pmatrix} e_2 & e_3 \\ f_2 & f_3 \end{pmatrix}$$

and distinguish two cases depending on the determinant.

On one hand, if $\det N \neq 0$, then as before, a change of variables let us write the verification equation (4) in the form

$$c_1 m + c_2 r + c_3 s + v = (r + w)s.$$

Since m must be used in the verification equation, we know that $c_1 \neq 0$. An easy calculation then shows that if (m, r, s) is a triple satisfying the verification equation for the keys v, w, then so does $(m - (c_2 - s)/c_1, r + 1, s)$. This gives us a forgery unless $c_2 = s$ for a non-negligible set of signatures. However, if this happens, then s would be a redundant signature element. From Lemma 48 we know that the scheme must be insecure.

On the other hand, if $\det N = 0$, we have the two cases $(e_2, e_3) = \lambda(f_2, f_3)$ or $(f_2, f_3) = 0$. If $(f_2, f_3) = 0$, then

$$\det \begin{pmatrix} c_2 & c_3 \\ e_2 & e_3 \end{pmatrix} \neq 0$$

or otherwise $(c_2, c_3), (e_2, e_3), (f_2, f_3)$ would be collinear, which would again allow us to compress the verification key. It follows that the verification equation (4) reduces to

$$r + v = (s + w)(f_1 m + f_4).$$

and since $f_1 \neq 0$, as the message must be used, we may query $m_1 = -f_1^{-1} f_4$ getting back a signature (r_1, s_1), where $r_1 = -v$. Now make another query with $m_2 = f_1^{-1}(1 - f_4)$ to get back a signature (r_2, s_2). Then

$$r_2 + v = s_2 + w \Rightarrow w = r_2 + v - s_2 = r_2 - r_1 - s_2,$$

so with two *chosen-message* queries the attacker can transfer V, W to $\mathbb{G}_2$ and then we know the scheme cannot be secure (concretely, $(R_1, R_1 R_2^{-1} S_2) = (V^{-1}, W^{-1})$ is a valid signature on any message). Therefore, we must have that $(e_2, e_3) = \lambda(f_2, f_3)$. Again using the fact that $(c_2, c_3), (e_2, e_3), (f_2, f_3)$ are not collinear, we must have

$$\det \begin{pmatrix} c_2 & c_3 \\ f_2 & f_3 \end{pmatrix} \neq 0.$$

It follows that we may do a change of variables $s' = f_1 m + f_2 r + f_3 s + f_4$ and $r' = c_1 m + c_2 r + c_3 s$ and by the collinearity of (e_2, e_3) and (f_2, f_3) the verification equation (4) becomes of the form

$$r + v = (e_1 m + e_3 s + w + e_6) s$$

and now if (m, r, s) is a valid signature, then so is $(m + 1/e_1, r + s, s)$, which is a valid forgery, since m must be used in the verification equation and hence $e_1 \neq 0$. Also, note that in the latter case, the attack can be performed in the random-message security game. □

4.4 SPS SCHEMES WITH VERIFICATION KEY $(V, W) \in \mathbb{G}_1 \times \mathbb{G}_2$

This section is devoted to the analysis of minimal Type II SPS schemes whose verification key is of the form $(V, W) \in \mathbb{G}_1 \times \mathbb{G}_2$. As mentioned in Section 4.3.2, Theorem 50 holds in this case even for EUF-RMA-secure schemes. This follows directly from Theorem 54 together with the lemma below.

Before going into the details of the proof, we give a short overview of the main proof ingredient, namely, how to prove the impossibility of a certain verification equation by using a computer algebra system. The proof below then consists of a large number of case distinctions. As a guide for choosing the right variable(s) for the basis of the case distinctions, we have used a Gröbner basis approach, which we describe next.

Using the notation from Section 4.3.3, when $(V, W) \in \mathbb{G}_1 \times \mathbb{G}_2$, the verification equation below can be written in the form

$$(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)(d_1 m + d_2 r + d_3 s + d_4 w + d_5) =$$
$$(e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_4 w + f_5).$$

We begin by defining the field

$$F = \mathbb{Q}(c_1, \ldots, c_6, d_1, \ldots, d_5, e_1, \ldots, e_6, f_1, \ldots, f_5).$$

To mount a random message attack, we assume that (m, r, s) is a random signature triple given to the adversary. In other words, we assume that (m, r, s) satisfies the verification equation. In the symbolic group model, the adversary will now have access to the elements $1, m, r, s$, so it is able to create elements of the form

$$\alpha_1 m + \alpha_2 r + \alpha_3 s + \alpha_4, \text{ where } \alpha_i \in \mathbb{Z}.$$

Therefore, any forgery that it can attempt to generate will be of the form

$$\begin{aligned} m^* &= m_1 m + m_2 r + m_3 s + m_4 \\ r^* &= r_1 m + r_2 r + r_3 s + r_4 \\ s^* &= s_1 m + s_2 r + s_3 s + s_4. \end{aligned}$$

Using this notation, we define

$$\begin{aligned} P_1 &= c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6 \\ P_2 &= d_1 m + d_2 r + d_3 s + d_4 w + d_5 \\ P_3 &= e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6 \\ P_4 &= f_1 m + f_2 r + f_3 s + f_4 w + f_5 \end{aligned}$$

and

$$\begin{aligned}
Q_1 &= c_1 m^* + c_2 r^* + c_3 s^* + c_4 v + c_5 w + c_6 \\
Q_2 &= d_1 m^* + d_2 r^* + d_3 s^* + d_4 w + d_5 \\
Q_3 &= e_1 m^* + e_2 r^* + e_3 s^* + e_4 v + e_5 w + e_6 \\
Q_4 &= f_1 m^* + f_2 r^* + f_3 s^* + f_4 w + f_5.
\end{aligned}$$

Now define

$$P = P_1 P_2 - P_3 P_4, \quad Q = Q_1 Q_2 - Q_3 Q_4.$$

Ideally, we would want to compute a Gröbner basis of the ideal generated by the coefficients of Q in the ring

$$F[m_1, \ldots, m_4, r_1, \ldots, r_4, s_1, \ldots, s_4]/(P),$$

but such a computation is out of reach of current CAS systems. Instead, we compute the polynomial $H = Q - P$. What we are a looking for is a non-trivial solution for the constants $m_1, \ldots, m_4$, $r_1, \ldots, r_4$ and $s_1, \ldots, s_4$ for which H will become the zero polynomial. Note that a trivial solution is always found by choosing

$$m_1 = 1, m_2 = m_3 = m_4 = 0, \quad r_2 = 1, r_1 = r_3 = r_4 = 0, \quad s_3 = 1, s_1 = s_2 = s_4 = 0,$$

which just means that we pick as the forgery attempt (m^*, r^*, s^*) the already given signature triple (m, r, s).

We can instead compute the Gröbner basis of the ideal I generated by the coefficients of H in the ring

$$F[m_1, \ldots, m_4, r_1, \ldots, r_4, s_1, \ldots, s_4]$$

When performing this computation, one of two things will happen:

1. $I = (m_1 - 1, m_2, m_3, m_4, r_1, r_2 - 1, r_3, r_4, s_1, s_2, s_3 - 1, s_4)$

2. I is some more complicated ideal.

If we get the former case, then we are unable to find a non-trivial solution by looking at the ideal I and we are forced to make a case distinction. We can try to pick e.g. c_1 and split into the cases $c_1 = 0$ and $c_1 \neq 0$. In the first case, we have to repeat the steps and in the second, we can scale the equation, so that $c_1 = 1$. This removes a constant from the equation and increases the likelihood of finding a workable Gröbner basis. In the latter case, we can read out a general formula for the forgery. Unfortunately, the formula might not always be valid. A typical problem is that the Gröbner basis contains a reciprocal of one of the coefficients in the field F. For example, if it contains c_1^{-1}, then we have to separately analyze the cases $c_1 = 0$ and $c_1 \neq 0$.

All in all, producing the full proof, will take a lot of trial and error, as well as CPU time, since for many equations, the Gröbner basis for the ideal I can take hours to compute. The problem also becomes choosing the right case distinctions that make the tree as small as possible. With the wrong choices, the proof can more than double in length.

Lemma 55. *An* EUF-RMA*-secure structure-preserving signature scheme in the Type II setting with* 1 *verification pairing-product equation,* 2 *group elements* $V \in \mathbb{G}_1$ *and* $W \in \mathbb{G}_2$ *in the verification key and* 2 *group elements in the signature requires at least* 3 *pairings in the pairing-product equation.*

Proof. We can write the verification equation in the form:

$$\begin{aligned}(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)(d_1 m + d_2 r + d_3 s + d_4 w + d_5) &= \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_4 w + f_5).&\end{aligned} \tag{5}$$

Our proof is a large case distinction based on the values of the parameters. The following Figure 3 illustrates the case distinction tree. We label the cases according to the tree and assume the reader follows the case distinctions by looking at the tree nodes.

Case 1. Assume first that $\det(d_2\ d_3; f_2\ f_3) \neq 0$. Then we can do a linear change of variables

$$\begin{pmatrix} r' \\ s' \end{pmatrix} = \begin{pmatrix} d_2 & d_3 \\ f_2 & f_3 \end{pmatrix} \begin{pmatrix} r \\ s \end{pmatrix} + \begin{pmatrix} d_1 m + d_4 w + d_5 \\ f_1 m + f_4 w + f_5 \end{pmatrix}.$$

$$\begin{tikzcd}
 & \text{root} \arrow[dl] \arrow[dr] & & & \\
1.\det\begin{pmatrix} d_2 & d_3 \\ f_2 & f_3 \end{pmatrix} \neq 0 & & 2.\det\begin{pmatrix} d_2 & d_3 \\ f_2 & f_3 \end{pmatrix} = 0 \arrow[dl] \arrow[dr] & & \\
 & 3.(d_2, d_3) = (0,0) \arrow[dl] \arrow[dr] & & 12.(d_2, d_3) \neq (0,0) & \\
4.\det\begin{pmatrix} d_1 & d_4 \\ f_1 & f_4 \end{pmatrix} \neq 0 & & 5.\det\begin{pmatrix} d_1 & d_4 \\ f_1 & f_4 \end{pmatrix} = 0 \arrow[dl] \arrow[dr] & & \\
 & 6.(d_1, d_4) = (0,0) & & 7.(d_1, d_4) \neq (0,0) \arrow[dl] \arrow[dr] & \\
 & & 8.d_1 \neq 0 \arrow[dl] \arrow[dr] & & 11.d_1 = 0 \\
 & 9.(f_2, f_3) = 0 & & 10.(f_2, f_3) \neq 0 &
\end{tikzcd}$$

FIGURE 3: Case distinction tree for the proof of Lemma 55.

After relabeling the constants and dropping the primes, this reduces the equation into the form

$$(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6) r = (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6) s$$

We can now choose $r = s = 0$, which is a valid signature for any m. Therefore, this case is impossible.

Case 2. We know that $\det(d_2\ d_3; f_2\ f_3) = 0$, so this splits into two subcases.

Case 3. If $(d_2, d_3) = 0$, then the verification equation (5) becomes

$$\begin{aligned} (c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)(d_1 m + d_4 w + d_5) = \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_4 w + f_5). \end{aligned} \tag{6}$$

We want to simplify the right factor on the left-hand side. This gives us two case distinctions.

Case 4. If $\det(d_2\ d_3; f_2\ f_3) \neq 0$, then we can make a change of variables $m' = d_1 m + d_4 w + d_5$, $w' = f_1 m + f_4 w + f_5$, which turns (6) into the form

$$\begin{aligned} (c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6) m = \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_2 r + f_3 s + w). \end{aligned} \tag{7}$$

If $f_2 \neq 0$, then $r = -f_2^{-1}w$, $s = 0$ is a valid signature for $m = 0$. If $f_3 \neq 0$, the same argument works with r and s swapped. Therefore, $f_2 = f_3 = 0$ and (7) becomes

$$(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)m = (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)w,$$

and in that equation, $(c_2, c_3), (e_2, e_3)$ cannot be collinear, as there would be only one independent signature element otherwise. As a result, a change of variables lets us rewrite it as:

$$(r + c_4 v)m = (s + e_4 v)w. \tag{8}$$

If $c_4 \neq 0$, then $r = s = 0$ gives a valid signature on $m = (e_4/c_4) \cdot w$. Therefore, we must have $c_4 = 0$, and for any valid message-signature pair $(m; r, s)$, the pair $(2m; r/2, s)$ is also valid, which breaks security against random-message attacks.

Case 5. $\det(d_2\ d_3; f_2\ f_3) = 0$, so either $(d_1, d_4) = 0$ or $(d_1, d_4) \neq 0$, i.e. $(f_1, f_4) = \lambda(d_1, d_4)$.

Case 6. If $(d_1, d_4) = 0$, then (6) becomes

$$\begin{aligned}(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)d_5 &= \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)&(f_1 m + f_2 r + f_3 s + f_4 w + f_5).\end{aligned}$$

If $d_5 = 0$, then verification equation requires just one pairing and this is easily shown to be impossible. Hence, $d_5 \neq 0$ and we may assume that $d_5 = 1$ by scaling. The verification equation thus becomes

$$\begin{aligned}c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6 &= \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)&(f_1 m + f_2 r + f_3 s + f_4 w + f_5).\end{aligned} \tag{9}$$

Assume now that $(m; r, s)$ is a valid message-signature pair satisfying the equation. Then $(m^*; r^*, s^*)$ given by:

$$\begin{aligned} m^* &= m + (e_2 f_3 - e_3 f_2)(f_1 m + f_2 r + f_3 s + f_4 w + f_5) + (c_3 f_2 - c_2 f_3) \\ r^* &= r + (e_3 f_1 - e_1 f_3)(f_1 m + f_2 r + f_3 s + f_4 w + f_5) + (c_1 f_3 - c_3 f_1) \\ s^* &= s + (e_1 f_2 - e_2 f_1)(f_1 m + f_2 r + f_3 s + f_4 w + f_5) + (c_2 f_1 - c_1 f_2) \end{aligned} \tag{10}$$

is another valid signature (easily obtained by looking for forgeries of the form $(m+\mu; r+\rho, s+\sigma)$ fixing the $\mathbb{G}_2$ side of (9), and solving the resulting linear system in (μ, ρ, σ). This is a forgery unless $m^* = m$ with overwhelming probability, which implies that

$$(e_2 f_3 - e_3 f_2)(f_1 m + f_2 r + f_3 s + f_4 w + f_5) + (c_3 f_2 - c_2 f_3) = 0,$$

and unless $\det(e_2\ e_3; f_2\ f_3) = \det(f_2\ f_3; c_2\ c_3) = 0$, that relation makes verification equation (9) linear, and hence the scheme clearly insecure.

Thus, we must have $\det(e_2\ e_3; f_2\ f_3) = \det(f_2\ f_3; c_2\ c_3) = 0$, and since the vectors $(c_2, c_3), (f_2, f_3), (e_2, e_3)$ cannot be all collinear by independence of the two signature elements, it follows that $(f_2, f_3) = 0$ and $\det(c_2\ c_3; e_2\ e_3) \neq 0$. Therefore, after a change of variables, (9) becomes:

$$r + c_4 v = (s + e_4 v)(f_1 m + f_4 w + f_5),$$

where $f_1 \neq 0$, since m must be used in the verification equation. Therefore, after $m' = f_1 m + f_4 w + f_5$, we get $r + c_4 v = (s + e_4 v) m$, which does not use w and must hence be insecure.

Case 7. In this case we have assumed that $\det(d_1\ d_4; f_1\ f_4) = 0$ and $(d_1, d_4) \neq 0$. It follows that $(f_1, f_4) = \lambda(d_1, d_4)$ and the verification equation is as in (6), i.e., it has the form

$$\begin{aligned}(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)(d_1 m + d_4 w + d_5) = \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_4 w + f_5).\end{aligned} \tag{11}$$

We now distinguish between whether d_1 is zero or not.

Case 8. Assume that $d_1 \neq 0$, so that we may make a change of variables $m' = d_1 m + d_4 w + d_5$, so that by $(f_1, f_4) = \lambda(d_1, d_4)$ the verification equation (11) becomes

$$\begin{aligned}(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6) m = \\ (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_5).\end{aligned} \tag{12}$$

namely we can drop the w term from the $\mathbb{G}_2$ side on the right. We now make a case distinction on whether $(f_2, f_3) = 0$.

Case 9. Assume that $(f_2, f_3) = 0$. In this case the equation becomes

$$(c_1m+c_2r+c_3s+c_4v+c_5w+c_6)m = (e_1m+e_2r+e_3s+e_4v+e_5w+e_6)(f_1m+f_5).$$

Now $\det(c_2\ c_3; e_2\ e_3) \neq 0$ or else there is just one independent signature element. It follows that we may simplify the equation to the form $(r + c_4v)m = (s + e_4v)(f_1m + f_5)$. However, w is not used here, so the scheme is trivially insecure.

Case 10. Assume that $(f_2, f_3) \neq 0$. Looking at equation (12), we see that $(c_2, c_3), (f_2, f_3), (e_2, e_3)$ cannot be all collinear or else there is just one independent signature element. It follows that one of $\det(c_2\ c_3; f_2\ f_3)$ and $\det(f_2\ f_3; e_2\ e_3)$ must be nonzero. In the first case, (12) becomes

$$(r + c_4v)m = (e_1m + e_2r + e_3s + e_4v + e_5w + e_6)s.$$

so that $m = 0, s = 0$ gives a valid signature for any r. In the latter case, (12) becomes

$$(c_1m + c_2r + c_3s + c_4v + c_5w + c_6)m = (r + e_4v)s,$$

which is insecure by the same argument.

Case 11. Since $d_1 = 0$, the equation (11) becomes

$$\begin{aligned}(c_1m + c_2r + c_3s + c_4v + c_5w + c_6)(d_4w + d_5) =\\ (e_1m + e_2r + e_3s + e_4v + e_5w + e_6)(f_1m + f_2r + f_3s + f_4w + f_5).\end{aligned}$$

Let $(m; r, s)$ be a valid message-signature pair, then so is $(m^*; r^*, s^*)$ given by:

$$\begin{aligned}
m^* &= m + (e_3f_2 - e_2f_3)(f_2r + f_3s + f_4w + f_5) + (c_2f_3 - c_3f_2)(d_4w + d_5)\\
r^* &= r + e_1f_2f_3r + e_1f_3^2s + (-c_1d_4f_3 + e_1f_3f_4)w - c_1d_5f_3 + e_1f_3f_5\\
s^* &= s - e_1f_2^2r - e_1f_2f_3s + (c_1d_4f_2 - e_1f_2f_4)w + c_1d_5f_2 - e_1f_2f_5
\end{aligned}$$

(obtained by the same approach as in Case 6). As in Case 6, this provides a valid forgery or lets us linearize the verification equation (concluding in both cases) unless the determinants $\det(e_2\ e_3; f_2\ f_3)$ and $\det(c_2\ c_3; f_2\ f_3)$ are both zero. In that case, since we know that $(c_2, c_3), (f_2, f_3), (e_2, e_3)$ can't be collinear, we obtain $(f_2, f_3) = 0$ and $\det(c_2\ c_3; e_2\ e_3) \neq 0$. This implies that we may write the verification equation in the form

$$(r + c_4 v)(d_4 w + d_5) = (s + e_4 v)(f_1 m + f_4 w + f_5).$$

The equation must depend on both m and w, so d_4 and f_1 are non zero, and we may simplify that further to $(r + c_4 v)w = (s + e_4 v)m$, and this has been ruled out in Case 4 already.

Case 12. We now have the original verification equation

$$(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)(d_1 m + d_2 r + d_3 s + d_4 w + d_5) =$$
$$(e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_3 s + f_4 w + f_5),$$

but we know that $(d_2, d_3) \neq 0$ and $(f_2, f_3) = \lambda(d_2, d_3)$. We note that the problem is completely symmetric with respect to d_2 and d_3, so we may assume that $d_2 \neq 0$. Set $r' = d_1 m + d_2 r + d_3 s + d_4 w + d_5$ and $s' = s$. Since $(f_2, f_3) = \lambda(d_2, d_3)$, the verification equation becomes

$$(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)r =$$
$$(e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_1 m + f_2 r + f_4 w + f_5).$$

If $f_1 \neq 0$, choose $r = 0$ and pick an m such that $f_1 m + f_4 w + f_5 = 0$. This is a valid forgery for any value of s. It follows that $f_1 = 0$, so that the equation becomes

$$(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6)r =$$
$$(e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6)(f_2 r + f_4 w + f_5).$$

Assume now that $(m; r, s)$ is a valid message-signature pair. Then so is $(m^*; r^*, s^*)$ given by

$$\begin{aligned} m^* &= m + (c_3 - e_3 f_2) r - e_3 f_4 w - e_3 f_5 \\ r^* &= r \\ s^* &= s + (e_1 f_2 - c_1) r + e_1 f_4 w + e_1 f_5. \end{aligned}$$

By the argument of Case 6 again, we deduce that the scheme is insecure (by forgery or linearization of the verification equation) unless perhaps if $c_3 - e_3 f_2 = 0$, $e_3 f_4 = 0$ and $e_3 f_5 = 0$. From $c_3 = e_3 f_2$, we see that if $e_3 = 0$, then $c_3 = 0$ and s would not be used in the verification equation, which is impossible. It follows that $e_3 \neq 0$ and $f_4 = f_5 = 0$. The verification equation is therefore

$$(c_1 m + c_2 r + c_3 s + c_4 v + c_5 w + c_6) r = (e_1 m + e_2 r + e_3 s + e_4 v + e_5 w + e_6) f_2 r.$$

It follows that $r = 0$ gives a forgery for any choice of m and s. Therefore, we have another contradiction and since we have exhausted all cases, the proof is complete. □

4.5 SYNTHESIS OF SPS

Our tool for the synthesis of SPS schemes consists of two components. The first component takes the description of a search space and generates all included SPS schemes. The second component classifies a given scheme by performing a proof and attack search.

4.5.1 *Generation of Schemes*

For our generation algorithm, we consider SPS schemes with generic KeyGen and Sign algorithms and assume all random values are sampled uniformly. Our definition of an SPS scheme consists of

- the employed group type and the supported message space $\mathbb{G}_1^k \times \mathbb{G}_2^l$, map $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. iso $\mathbb{G}_2 \to \mathbb{G}_1$.

input $[V]$ in $\mathbb{G}_1$. input $[W]$ in $\mathbb{G}_2$.

oracle $o(M : \mathbb{G}_2) =$ sample R; return $[R, (1 + W^2 + M * V) * R^{-1}]$ in $\mathbb{G}_2$.

win $(M' : \mathbb{G}_2, R' : \mathbb{G}_2, S' : \mathbb{G}_2) = (S' * R' = 1 + W^2 + V * M' \wedge \forall i : M' \neq M_i)$.

FIGURE 4: Example of input for analyzing EUF-CMA security of an SPS scheme.

- the randomly sampled values $u_i \in \mathbb{Z}_p$ used in KeyGen,
- the verification keys $V_i = G^{f_i(\vec{u})} \in \mathbb{G}_1$ and $W_i = H^{g_i(\vec{u})} \in \mathbb{G}_2$,
- the randomly sampled values $r_i \in \mathbb{Z}_p$ used in Sign,
- the signature elements $S_i = G^{s_i(\vec{u},\vec{r},\vec{m})} \in \mathbb{G}_1$ and $T_i = H^{t_i(\vec{u},\vec{r},\vec{m})} \in \mathbb{G}_2$, and
- the pairing-product equations used by Verify.

Here, f_i and g_i are arbitrary rational functions in the random variables $\vec{u}$. Similarly, s_i and t_i are rational functions in the random variables $\vec{u}, \vec{r}$ and the discrete logarithms $\vec{m}$ of the messages such that there exists a corresponding generic signing algorithm, i.e., S_i and T_i can be computed without knowing the discrete logarithms of the messages.

A search space description characterizes a finite set of SPS schemes and consists of (1) the group type, (2) the number of messages, verification key elements, and signature elements in $\mathbb{G}_1$ and $\mathbb{G}_2$, (3) the number of random values sampled in KeyGen and Sign, and (3) a description of the rational expressions that can be used for f_i, g_i, s_i, and t_i.

There are two ways to characterize the allowed rational expressions. First, the tool can take a set of Laurent polynomials with placeholders and allowed values for these placeholders, and generate all instances. Second, the tool accepts a set of constraints that specify bounds on the number of additions, the size of coefficients, and the degree of monomials. Then, it generates all Laurent polynomials that satisfy these constraints.

Given a search space description and concrete polynomials for the verification keys and the signature elements, the tool can compute the (strongest) verification equation as follows. Using distinct variables $Z_1, Z_2, \dots$ for all group elements in the verification keys, signature elements, and messages, enumerate all products over these variables that can be computed by applying the homomorphisms and the bilinear map. This yields a sequence of monomials $M_1, M_2, \dots$ over the the variables Z_i denoting products in $\mathbb{G}_T$ that can be computed from the input of the verification algorithm using Ψ and e. To characterize the linear relations between the elements in $\mathbb{G}_T$ corresponding to the monomials M_i, we associate a rational expression F_i over $\vec{u}, \vec{r}, \vec{m}$ to M_i by evaluating the monomial for $Z_i := h_i(\vec{u}, \vec{r}, \vec{m})$ where h_i is the exponent of the group element associated with Z_i. Finally, we use linear algebra to compute a basis of the linear relations between the F_i and map them back to verification equations using M_i.

4.5.2 *Proof and Attack Search*

We classify generated schemes using a proof and attack search based on an extension of the Generic Group Analyzer (GGA). To analyze SPS schemes, we use GGA's support for the generic bilinear group model. The definition in Figure 4 specifies the EUF-CMA security experiment for an SPS scheme in the Type II setting. The verification keys are specified in the second line, the signing algorithm is given in the third line, and the winning condition (including the verification equation) is given in the last line. Here, group elements are specified by giving their exponent polynomials and the variables V and W are assumed to be randomly sampled. For such an input and a bound on the number q of performed oracle queries, the tool either returns an attack or a proof that the scheme is q-EUF-CMA secure in the generic group model.

4.5.3 *Synthesized Schemes*

We have performed an exhaustive search for Type II schemes with keys $V, W \in \mathbb{G}_1$, message $M \in \mathbb{G}_2$, and signature $T, S \in \mathbb{G}_2$ such that: V and W are random;

Search Space		Schemes		Results (for eq. classes)		
Verification equation	First signature element	total	eq. cl.	Noverif	Attack	Proof
$s = f(r, v, w, m)$	$T = H^r$ for random r	212	57	0	55	1
$s\,t = f(r, v, w, m)$	$T = H^r$ for random r	224	67	0	55	12
$s\,(t - w) = f(r, v, w, m)$	$T = H^{r+w}$ for random r	1344	774	651	103	14
$s\,w = f(r, v, w, m)$	$T = H^r$ for random r	224	126	0	120	3
		2004	1024	651	333	30

TABLE 2: Synthesis results for Type II with keys $V, W \in \mathbb{G}_1$, message $M \in \mathbb{G}_2$, and signature $T, S \in \mathbb{G}_2$.

$T = H^r \cdot U$ where r is random and U does not involve r; the exponent polynomials of S, i.e., $s(r, v, w, m)$, have coefficients in $\{0, 1\}$. The results of our search are presented in Table 2. We use "Proof" to denote that our tool could prove at least 2-EUF-CMA security. Among the SPS schemes that are found, we identify equivalent schemes according to the following notion: we say two schemes Σ and Σ' are in the same equivalence class if Σ can be obtained from Σ' by applying invertible affine transformations to the verification keys, the messages, and the signature elements. This implies the existence of reductions from the security of Σ to the security of Σ' and vice-versa. As a simple example, consider a scheme that is obtained from another scheme by first multiplying the message M with G and then applying the original signing algorithm.

4.5.4 *New SPS Schemes*

Among the schemes that we found using our tool, we highlight two of them that are of particular interest, as well as a counterpart of the first one in the Type III setting.

A Strongly-Optimal Randomizable SPS.

The first scheme is an SPS which is randomizable and matches the lower bound of Theorem 50, i.e., it can be verified using one offline and two online pairings. This scheme improves over the previously known randomizable schemes (in particular over the one recently proposed in [AGOT14a]) as the latter requires three online pairings. This new scheme is presented in Fig. 5 with its security proven below.

Theorem 56. *The signature scheme in Fig. 5 is* EUF-CMA-*secure in the generic bilinear group model.*

Proof. Since we are in the generic bilinear group model, the adversary $\mathcal{A}$ can only perform generic operations via the oracles. Thus, in $\mathbb{G}_1$ and $\mathbb{G}_2$ it can compute only linear combinations of elements that are in the public key, or elements that were returned by the signing oracle (i.e., signatures). Such linear combinations in $\mathbb{G}_2$ correspond to formal Laurent polynomials whose variables are the discrete logarithms of the corresponding group elements.

From the public key, $\mathcal{A}$ sees in $\mathbb{G}_1$ the elements G, V, W which correspond to $1, v, w$ respectively, while in $\mathbb{G}_2$ $\mathcal{A}$ only sees H, which corresponds to 1. For every signing query M_i (with discrete log m_i), the adversary receives $R_i = H^{r_i}, S_i = M_i^{v/r_i} H^{w/r_i} \in \mathbb{G}_2$, namely r_i and $s_i = m_i v / r_i + w / r_i$.

Let $(M, R, S) \in \mathbb{G}_2^3$ be the forgery returned by the adversary after making q signing queries. From the assumption that $\mathcal{A}$ is generic, these three elements in $\mathbb{G}_2$ can be computed only as linear combinations of all the available material in $\mathbb{G}_2$. Namely, when looking at their discrete logarithms (m, r, s), it must be

$$\begin{aligned}
m &= \mu + \sum_{i=1}^{q} \mu_{r_i} r_i + \sum_{i=1}^{q} \mu_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i} \right) \\
r &= \rho + \sum_{i=1}^{q} \rho_{r_i} r_i + \sum_{i=1}^{q} \rho_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i} \right) \\
s &= \sigma + \sum_{i=1}^{q} \sigma_{r_i} r_i + \sum_{i=1}^{q} \sigma_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i} \right)
\end{aligned}$$

Moreover, in order for the adversary to succeed in the EUF-CMA security game, the verification equation

$$rs = mv + w$$

must be satisfied with significant probability over the random choices of v, w and all the r_i sampled in the security experiment. By Schwartz-Zippel, Lemma 17, the above equation holds as an identity of polynomials.

Therefore, in our proof we show that for every forgery (m, r, s) which verifies correctly it must be the case that $\mathcal{A}$ reuses one of the previously obtained messages, i.e., $m = m_j$ for some $j \in [q]$.

The verification equation is $rs = mv + w$ that, by expanding r, s we can write as

$$\left(\rho + \sum_{i=1}^{q} \rho_{r_i} r_i + \sum_{i=1}^{q} \rho_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) \cdot \left(\sigma + \sum_{i=1}^{q} \sigma_{r_i} r_i + \sum_{i=1}^{q} \sigma_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w \tag{13}$$

First, we show that it must be $\sigma_{r_i} = 0$ for all $i = 1, \ldots, q$. Assume by contradiction that $\exists j \in [q] : \sigma_{r_j} \neq 0$. Then when looking at the coefficients of r_j^2, these must be zero, i.e., $\sigma_{r_j} \rho_{r_j} = 0$ which implies $\rho_{r_j} = 0$. Similarly, when analyzing the monomials $r_i r_j$ for all $i \neq j$, their coefficients $\rho_{r_i} \sigma_{r_j} + \rho_{r_j} \sigma_{r_i}$ must be zero. Since $\rho_{r_j} = 0$ and $\sigma_{r_j} \neq 0$ it must be $\rho_{r_i} = 0$ for all $i \neq j$. Hence, so far we have shown that it must be the case that $\rho_{r_i} = 0 \ \forall i \in [q]$.

If we now look at the coefficient of r_j (on the left-hand side of equation (13)), i.e., $\rho\sigma_{r_j}$, this must be zero which implies $\rho = 0$. Essentially, equation (13) can be rewritten as

$$\left(\sum_{i=1}^{q} \rho_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) \cdot \left(\sigma + \sum_{i=1}^{q} \sigma_{r_i} r_i + \sum_{i=1}^{q} \sigma_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w \tag{14}$$

Now, let us analyze the terms w/r_i whose coefficients must be zero as well, i.e., $\sigma\rho_{s_i} = 0 \ \forall i \in [q]$. This shows that $\sigma = 0$, otherwise (if $\sigma \neq 0$) it must be the case that $\rho_{s_i} = 0 \ \forall i \in [q]$, thus yielding a zero polynomial on the left-hand side of equation (14) which is impossible.

If we then look at the coefficients of terms $\frac{w r_j}{r_i}$ for all $i \neq j$, these must be zero, i.e., $\rho_{s_i}\sigma_{r_j} = 0$, which means $\rho_{s_i} = 0$ for all $i \neq j$ since $\sigma_{r_j} \neq 0$ by our assumption. Thus equation (14) can be rewritten as

$$\rho_{s_j}\left(\frac{m_j v}{r_j} + \frac{w}{r_j}\right) \cdot \left(\sum_{i=1}^{q} \sigma_{r_i} r_i + \sum_{i=1}^{q} \sigma_{s_i}\left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w \qquad (15)$$

where $\rho_{s_j} \neq 0$ otherwise there would be a zero polynomial on the left-hand side of the equation.

Now, if we look at the terms $\frac{w^2}{r_i r_j}$, using a similar argument as above we obtain that $\sigma_{s_i} = 0$ for all $i \in [q]$. And if we look at the terms $\frac{w r_k}{r_j}$ for all $k \neq j$, it must be $\sigma_{r_k}\rho_{s_j} = 0$, which means $\sigma_{r_k} = 0$ for all $k \neq j$.

Hence, we are left with $\rho_{s_j}\left(\frac{m_j v}{r_j} + \frac{w}{r_j}\right)\sigma_{r_j} r_j = mv + w$ which simplifies to $\rho_{s_j}\sigma_{r_j}(m_j v + w) = mv + w$ that forces $\rho_{s_j}\sigma_{r_j} = 1$ and thus $m_j = m$, which is a contradiction since (m, r, s) is a forgery only if $m \neq m_i$ for all $i = 1, \ldots, q$.

Therefore, it must be the case that $\sigma_{r_i} = 0$ for all $i = 1, \ldots, q$.

By using very similar arguments to the ones presented above, we can also argue that $\sigma_{r_i} = 0$ for all $i = 1, \ldots, q$ unless $m = m_j$.

Hence, since $\rho_{r_i} = \sigma_{r_i} = 0\ \forall i \in [q]$, equation (13) can be rewritten as

$$\left(\rho + \sum_{i=1}^{q} \rho_{s_i}\left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) \cdot \left(\sigma + \sum_{i=1}^{q} \sigma_{s_i}\left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w \qquad (16)$$

Now, if we expand m in the right-hand side of the above equation, one can see that when considering the formal polynomials the equation cannot hold. To see this, assume that $\exists j : \rho_{s_j} \neq 0$. By looking at the coefficients of $\frac{w^2}{r_i r_j}$, it must hold $\sigma_{s_i} = 0$ for all $i \in [q]$. It is clear that $\sigma \neq 0$ otherwise one has a zero polynomial on the left-hand side of equation (16). At this point either $mv + w$ contains the constant term $\rho\sigma$, or (if $\rho = 0$) it must contain a term w/r_j which is not the case either. This shows that $\rho_{s_i} = 0\ \forall i \in [q]$. Analogously, one can see that equation $\rho\left(\sigma + \sum_{i=1}^{q} \sigma_{s_i}\left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w$ cannot hold as well. $\square$

This scheme can be translated to Type III groups using the transformation in [CM14], which essentially consists in "duplicating" R, i.e., to give $R = G^r \in \mathbb{G}_1$

$\mathrm{Setup}(1^k)$: return $\mathbb{P} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow \mathcal{G}(1^k)$.

$\mathrm{KeyGen}(\mathbb{P})$: choose random $v, w \leftarrow \mathbb{Z}_p$ and return $VK = (V, W)$, $SK = (v, w)$ where $V = G^v$ and $W = G^w$.

$\mathrm{Sign}(\mathbb{P}, SK, M)$: given $M \in \mathbb{G}_2$, choose a random $r \leftarrow \mathbb{Z}_p^*$ and return (R, S) where $R = H^r$ and $S = (M^v H^w)^{1/r}$.

$\mathrm{Rerand}(\mathbb{P}, VK, M, (R, S))$: pick a random $\alpha \leftarrow \mathbb{Z}_p^*$ and compute a randomized signature (R', S') as $R' = R^\alpha$ and $S' = S^{1/\alpha}$.

$\mathrm{Verify}(\mathbb{P}, VK, M, (R, S))$: accept if and only if $M, R, S \in \mathbb{G}_2$ and

$$e(\psi(R), S) = e(V, M) \cdot e(W, H).$$

FIGURE 5: Our strongly-optimal re-randomizable SPS.

and $T = H^r \in \mathbb{G}_2$, and adding a pairing-product equation to check that R, T have the same discrete logarithm, i.e., $e(R, H) = e(G, T)$. Such transformed scheme however requires one offline and four online pairings in the pairing-product equations. In what follows we propose a slightly different way to transform our scheme in the Type III setting which yields a solution requiring only three online pairings. The basic idea is that in the previous transformation T is not used in the first pairing-product equation, and its utility is to force the adversary to show that it knows the discrete log of R (or obtained (R, T) by applying a linear operation on a pair received by the challenger). We obtain the same functionality by letting the signer compute $T = H^{1/r}$. This allows us to test "equality of r between R and T" by checking $e(R, T) = e(G, H)$. The last pairing, however, involves only the generators and can thus be computed offline. A precise description of the resulting scheme is provided in Fig. 6, and the security is proven by the following theorem.

Theorem 57. *The signature scheme in Fig. 6 is* EUF-CMA*-secure in the generic bilinear group model.*

Proof. The proof proceeds similarly to the one of Theorem 56 except that now we have to take into considerations the two verification equations as well as the additional information which is received in the different groups.

From the public key, $\mathcal{A}$ sees in $\mathbb{G}_1$ the elements G, V, W which correspond to discrete logs $1, v, w$ respectively, while in $\mathbb{G}_2$ $\mathcal{A}$ only sees H, which corresponds to 1. For every signing query M_i (with discrete log m_i), the adversary receives $R_i = G^{r_i} \in \mathbb{G}_2, T_i = H^{1/r_i}, S_i = M_i^{v/r_i} H^{w/r_i} \in \mathbb{G}_2$, namely r_i, $1/r_i$ and $s_i = m_i v/r_i + w/r_i$.

Let $(M, R, T, S) \in \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_2^2$ be the forgery returned by the adversary after making q signing queries. From the assumption that $\mathcal{A}$ is generic, the elements (M, R, S) in $\mathbb{G}_2$ can be computed only as linear combinations of all the available material in $\mathbb{G}_2$, and similarly T can be computed only as a linear combination of the available group elements in $\mathbb{G}_1$. Namely, when looking at their discrete logarithms (m, r, t, s), it must be

$$\begin{aligned}
m &= \mu + \sum_{i=1}^{q} \mu_{t_i} \frac{1}{r_i} + \sum_{i=1}^{q} \mu_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i} \right) \\
r &= \rho + \sum_{i=1}^{q} \rho_{r_i} r_i + \rho_v v + \rho_w w \\
t &= \tau + \sum_{i=1}^{q} \tau_{t_i} \frac{1}{r_i} + \sum_{i=1}^{q} \tau_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i} \right) \\
s &= \sigma + \sum_{i=1}^{q} \sigma_{t_i} \frac{1}{r_i} + \sum_{i=1}^{q} \sigma_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i} \right)
\end{aligned}$$

Moreover, in order for the adversary to succeed in the EUF-CMA security game, the two verification equations

$$rs = mv + w \wedge rt = 1$$

must be satisfied with significant probability over the random choices of v, w and all the r_i sampled in the security experiment. By Schwartz-Zippel, Lemma 17, the above equations hold as identities of polynomials.

For our proof we first consider the second equation $rt = 1$ and we show that, when expanding r, t as follows

$$\left(\rho + \sum_{i=1}^{q} \rho_{r_i} r_i + \rho_v v + \rho_w w \right) \cdot \left(\tau + \sum_{i=1}^{q} \tau_{t_i} \frac{1}{r_i} + \sum_{i=1}^{q} \tau_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i} \right) \right) = 1 \qquad (17)$$

it must be the case that both ρ_v and ρ_w are 0.

Assume by contradiction that $\rho_w \neq 0$. Then by looking at the coefficients of w^2/r_i, these must be zero, which implies that $\tau_{s_i} = 0$ for all $i \in [q]$. Similarly, we also obtain that all coefficients of w/r_i must be zero, and thus $\tau_{t_i} = 0, \forall i \in [q]$. So, we are left with

$$\left(\rho + \sum_{i=1}^{q} \rho_{r_i} r_i + \rho_v v + \rho_w w\right) \cdot \tau = 1 \tag{18}$$

which, being $\rho_w \neq 0$ is however impossible. Hence, $\rho_w = 0$, and in an analogous way, one can prove that also $\rho_v = 0$.

In conclusion, from the fact that the forgery satisfies $rt = 1$ we obtain that it must be the case that $\rho_v = \rho_w = 0$. In the next part of the proof we will use this information to derive a contradiction on the fact that $m \neq m_i$ for all $i = 1$ to q.

Plugging this information in the first verification equation expanded for r, s we obtain

$$\left(\rho + \sum_{i=1}^{q} \rho_{r_i} r_i\right) \cdot \left(\sigma + \sum_{i=1}^{q} \sigma_{t_i} \frac{1}{r_i} + \sum_{i=1}^{q} \sigma_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w \tag{19}$$

First, we show that $\sigma_{t_i} = 0$ for all $i = 1, \ldots, q$. Indeed, assume by contradiction that $\exists j : \sigma_{t_j} \neq 0$. Then, by looking at the coefficients of terms r_i/r_j for all $i \neq j$ we obtain that $\rho_{r_i} = 0$ for all $i \neq j$. And similarly, since the term $1/r_j$ is not in $mv + w$ it must be $\rho\sigma_{t_j} = 0$ and thus $\rho = 0$.

Essentially, we can rewrite equation (19) as

$$\rho_{r_j} r_j \cdot \left(\sigma + \sum_{i=1}^{q} \sigma_{t_i} \frac{1}{r_i} + \sum_{i=1}^{q} \sigma_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w \tag{20}$$

where $\rho_{r_j} \neq 0$ otherwise we would get a zero polynomial on the left-hand side of the equation above. By analyzing the terms r_j/r_i for all $i \neq j$ we obtain that $\sigma_{t_i} = 0$ for all $i \neq j$. Similarly, we also have $\sigma = 0$. When looking at the terms

wr_j/r_i for all $i \neq j$, we obtain that $\sigma_{s_i} = 0$ for all $i \neq j$. This allows us to rewrite equation (20) as

$$\rho_{r_j} r_j \cdot \left(\sigma_{t_j} \frac{1}{r_j} + \sigma_{s_j} \left(\frac{m_j v}{r_j} + \frac{w}{r_j}\right)\right) = mv + w$$

which simplifies to

$$\rho_{r_j} \sigma_{s_j} \left(m_j v + w\right) = mv + w \tag{21}$$

since σ_{t_j} must be zero. However, equation (21) holds only if $m = m_j$ which is a contradiction. Therefore, $\sigma_{t_i} = 0$ for all $i = 1, \ldots, q$.

Using this information, we can rewrite equation (19) as

$$\left(\rho + \sum_{i=1}^{q} \rho_{r_i} r_i\right) \cdot \left(\sigma + \sum_{i=1}^{q} \sigma_{s_i} \left(\frac{m_i v}{r_i} + \frac{w}{r_i}\right)\right) = mv + w \tag{22}$$

Since the terms $r_j w/r_i$ must have coefficients zero, it must be $\rho_{r_j} \sigma_{s_i} = 0$ for all $i \neq j$. We claim that there exists exactly one $k \in [q]$ such that both ρ_{r_k} and σ_{s_k} are non-zero while $\rho_{r_i} = \sigma_{s_i} = 0$ for all $i \neq k$. First, assume by contradiction that there exists two distinct indices k_1, k_2 such that $\rho_{r_{k_1}} \neq 0$ and $\sigma_{s_{k_2}} \neq 0$. Then it would be the case the term wr_{k_1}/r_{k_2} has a non-zero coefficient, which is impossible. Second, we show that it cannot be the case that only one of ρ_{r_k} and σ_{s_k} is non-zero. Indeed assume by contradiction that $\rho_{r_k} = 0$, then one can see that the equation (22) cannot be satisfied. Similarly, it holds $\sigma_{s_k} \neq 0$.

Therefore, it must be the case that $\rho_{r_k} \neq 0$ and $\sigma_{s_k} \neq 0$, and we can simplify the equation as

$$(\rho + \rho_{r_k} r_k) \cdot \left(\sigma + \sigma_{s_k} \left(\frac{m_k v}{r_k} + \frac{w}{r_k}\right)\right) = mv + w \tag{23}$$

However, it is not hard to see that it must be $\rho = \sigma = 0$, which means $\rho_{r_k} \sigma_{s_k} (m_k v + w) = mv + w$, i.e., $m = m_k$, which is a contradiction. □

A Strongly-Optimal RMA-Secure SPS.

Our second new SPS scheme, presented in in Fig. 7, is secure against random-message attacks, and achieves the lower bound of only two pairings in the

$\mathrm{Setup}(1^k)$: return $\mathbb{P} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, H) \leftarrow \mathcal{G}(1^k)$.

$\mathrm{KeyGen}(\mathbb{P})$: choose random $v, w \leftarrow \mathbb{Z}_p$ and return $VK = (V, W)$, $SK = (v, w)$ where $V = G^v$ and $W = G^w$.

$\mathrm{Sign}(\mathbb{P}, SK, M)$: given $M \in \mathbb{G}_2$, choose a random $r \leftarrow \mathbb{Z}_p^*$ and return $(R, T, S) \in \mathbb{G}_1 \times \mathbb{G}_2^2$ where $R = G^r$, $T = H^{1/r}$ and $S = (M^v H^w)^{1/r}$.

$\mathrm{Rerand}(\mathbb{P}, VK, M, (R, T, S))$: pick a random $\alpha \leftarrow \mathbb{Z}_p^*$ and compute a randomized signature (R', T', S') as $R' = R^\alpha$, $T' = T^{1/\alpha}$ and $S' = S^{1/\alpha}$.

$\mathrm{Verify}(\mathbb{P}, VK, M, (R, T, S))$: accept if and only if $R \in \mathbb{G}_1$, $M, T, S \in \mathbb{G}_2$ and

$$e(R, S) = e(V, M) \cdot e(W, H) \quad \text{and} \quad e(R, T) = e(G, H).$$

FIGURE 6: A re-randomizable SPS in Type III groups.

pairing-product equation (both necessarily online) for EUF-RMA-secure schemes, as stated in Theorem 54. In particular, it *beats* the lower bound of Theorem 50 that holds for EUF-CMA-secure schemes.

This scheme is also perfectly randomizable, with the simple randomization algorithm that sends a signature (R, S) on M to $(R \cdot H^t, S \cdot M^t)$ for some uniformly random t.

As an interesting note, we observe that the verification equation of this scheme is exactly the only possible one, according to our impossibility proof. Indeed, while our Lemma 52 holds for SPS schemes that are EUF-CMA-secure, the actual proof relies on EUF-RMA-security in all cases but one. For that particular case, in which we show a chosen-message attack, the verification equation is of the form $s + w = (r + v)(f_1 m + f_4)$ for some constants f_1, f_4, up to invertible linear transformations on the verification key and signature elements.

Theorem 58. *The signature scheme in Fig. 7 is* EUF-RMA-*secure in the generic bilinear group model.*

Proof. In this case, the generic adversary receives the verification key $V, W \in \mathbb{G}_1$ as well as q valid message-signature pairs $(M_i; R_i, S_i) \in \mathbb{G}_2 \times \mathbb{G}_2^2$, $i = 1, \ldots, q$, on random messages In particular, the discrete logarithms $(m_i; r_i, s_i)$ satisfy that

m_i, r_i are uniformly random in $\mathbb{Z}_p$ and s_i is given by $s_i = (r_i + v)m_i - w$. We need to prove that this information does not enable the adversary to construct a valid signature (R, S) on any message M other the M_i's themselves.

Suppose that he does construct a valid pair $(M; R, S)$, with discrete logarithms $(m; r, s)$. Since the adversary is generic and the only available elements in $\mathbb{G}_2$ are H and the M_i's, R_i's and S_i's, m, r, s must be explicit linear combinations of $1, m_1, \dots, m_q, r_1, \dots, r_q, s_1, \dots, s_q$. In other words, m, r, s are of the form:

$$\begin{aligned} m &= \mu + \sum_{i=1}^{q} \mu_{m_i} m_i + \mu_{r_i} r_i + \mu_{s_i}\big((r_i + v)m_i - w\big), \\ r &= \rho + \sum_{i=1}^{q} \rho_{m_i} m_i + \rho_{r_i} r_i + \rho_{s_i}\big((r_i + v)m_i - w\big), \\ s &= \sigma + \sum_{i=1}^{q} \sigma_{m_i} m_i + \sigma_{r_i} r_i + \sigma_{s_i}\big((r_i + v)m_i - w\big). \end{aligned}$$

Moreover, for the adversary to succeed, the verification equation

$$rm + vm = s + w \tag{24}$$

should hold with significant probability on the choice of the private key v, w and the random values m_i, r_i. As a result, the Schwartz-Zippel lemma [? ? ?] implies that relation (24) holds as an identity of polynomials in $\mathbb{Z}_p[v, w, m_1, \dots, m_q, r_1, \dots, r_q]$.

Now, the monomial v appears only in the term vm, with coefficient μ. Therefore, $\mu = 0$. Similarly, the monomials vr_i and $v^2 m_i$ appear only in vm, with coefficient $\mu_{r_i} + \mu_{s_i}$ and μ_{s_i} respectively. Therefore, $\mu_{r_i} = \mu_{s_i} = 0$ for all i, and m simply becomes:

$$m = \sum_{i=1}^{q} \mu_{m_i} m_i.$$

Clearly, the coefficients μ_{m_i} cannot all be zero: otherwise, we obtain $s + w = 0$, and inspection of the coefficients of the monomials 1, m_i, r_i and vm_i yields that all the coefficients of s vanish, and hence $w = 0$, which is a contradiction.

Setup(1^k): return $\mathbb{P} = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, \psi, G, H) \leftarrow \mathcal{G}(1^k)$.

KeyGen$(\mathbb{P})$: choose random $v, w \leftarrow \mathbb{Z}_p$ and return $VK = (V, W)$, $SK = (v, w)$ where $V = G^v$ and $W = G^w$.

Sign$(\mathbb{P}, SK, M)$: given $M \in \mathbb{G}_2$, choose a random $r \leftarrow \mathbb{Z}_p$ and return (R, S) where $R = H^r$ and $S = M^{r+v} H^{-w}$.

Verify$(\mathbb{P}, VK, M, (R, S))$: accept if and only if $M, R, S \in \mathbb{G}_2$ and

$$e(\psi(S) \cdot W, H) = e(R \cdot V, M).$$

FIGURE 7: Our RMA-secure SPS with two pairings.

Therefore, we can fix an index j such that $\mu_{m_j} \neq 0$. The monomial m_j^2 appears only in the term rm with coefficient $\mu_{m_j}\rho_{m_j}$, hence $\rho_{m_j} = 0$, and $m_i m_j$, $i \neq j$ also appears only in rm with coefficient $\mu_{m_i}\rho_{m_j} + \mu_{m_j}\rho_{m_i} = \mu_{m_j}\rho_{m_i}$, hence $\rho_{m_i} = 0$ for all i. Similarly, inspection of the coefficients of $vr_i m_j$, $r_i m_j$ yields $\rho_{s_i} = 0$ for all i and $\rho_{r_i} = 0$ for all $i \neq j$ respectively. Hence:

$$r = \rho + \rho_{r_j} r_j.$$

Then, inspection of the coefficients of vm_i and $r_i m_i$ gives $\sigma_{s_i} = \mu_{m_i} = \mu_{m_i}\rho_{r_i}$ for all i. In particular, we get $\sigma_{s_j} = \mu_{m_j}$, $\rho_{r_j} = 1$, and for all $i \neq j$, $\sigma_{s_i} = \mu_{m_i} = 0$. Moreover, the coefficient of w is 0 on the left-hand side and $1 - \sum_i \sigma_{s_i}$ on the right-hand side, hence $\sigma_{s_j} = \mu_{m_j} = 1$. This shows that $m = m_j$, which concludes the proof. $\square$

5 SUMMARY OF CONTRIBUTIONS

In this chapter we briefly state our main original contributions.

5.1 GENERALIZING THE GENERIC GROUP MODEL

We introduce the concept of a group setting to describe recent constructions in the literature, e.g. leveled multilinear maps. Second, we extend Maurer's definition of the generic group model to handle any group setting. Third, we define a very general framework, generalized extraction problems, for expressing many complicated forms of cryptographic assumptions and definitions. In particular, in the setting of interactive assumptions, generalized extraction problems allow us to describe complicated conditions, such as the winning condition under a chosen message attack for a structure-preserving signature scheme.

5.2 MASTER THEOREMS FOR THE EXTENDED GGM

In Chapter 3 we define a symbolic group model corresponding to a group setting and relate the distinguishing probability of a symbolic group model oracle with a corresponding generic group model oracle under a polynomially induced distribution providing the initial internal state. Furthermore, we prove a master theorem with a machine-checkable side-condition that relates the distinguishing probability of two generic group models having their initial state sampled through two different polynomially induced distributions. Our master theorem encapsulates and extends all previous master theorems stated in the literature. Finally, we prove a novel type of master theorem, an interactive master theorem, that not only encapsulates generalized extraction problems, but also allows analyzing interactive assumptions. Interactive assumptions are particularly tricky to analyze by hand and in the literature there has been several incorrect proofs in the interactive setting that have passed peer review.

5.3 AUTOMATED ANALYSIS IN THE EXTENDED GGM

We identify three broad types of cryptographic assumptions: non-parametric, parametric and interactive. We develop algorithms for checking the side-conditions of our master theorems for all three assumption types. In the non-parametric and parametric setting the side-condition is a problem of linear algebra. When the assumption is parametric, the side-condition is, unfortunately, a problem involving matrices of variable dimension, which typically can't be handled by computer algebra systems. We identify a restricted category of parametric assumptions, where we show that solving the linear algebra problem is equivalent to solving a system of equations. Even in this restricted case, we show that solving the system of equations is undecidable. However, we also discover that SMT solvers are able to find solutions or prove their absence for the resulting system of equations that arise from typical cryptographic assumptions. Finally, in the interactive setting, we develop an algorithm that uses Gröbner basis techniques as well as SMT solvers to check the side-condition of the interactive master theorem. All these algorithms are real-world tested by having been implemented in a practical tool, the Generic Group Analyzer.

5.4 SYNTHESIS OF SPS AND NEW SPS SCHEMES

Using the interactive solver of our Generic Group Analyzer, we develop a template system that allows us to specify families of structure-preserving signature (SPS) schemes for which we can automatically generate input scripts for the Generic Group Analyzer. This allows us to generate large number of candidate SPS schemes matching certain design criteria in order to find new SPS schemes through brute force. As a result of our brute-force search, we identify two new Type II SPS schemes that improve on the most efficient previously known ones.

5.5 NEW LOWER BOUNDS FOR SPS

Through our brute-force search of Type II SPS schemes, we are able to conjecture lower bounds for SPS schemes in the Type II setting with respect to the number of pairing computations required in the verification equation of a signature. We then proceed to prove by hand that the lower bounds hold. Combined with the synthesis results, we are able to show that our lower bounds for Type II SPS schemes are tight, in other words, our synthesized schemes are optimal.

BIBLIOGRAPHY

[ACdM05] Giuseppe Ateniese, Jan Camenisch, and Breno de Medeiros. Untraceable RFID tags via insubvertible encryption. In Vijayalakshmi Atluri, Catherine Meadows, and Ari Juels, editors, *ACM CCS 05*, pages 92–101. ACM Press, November 2005.

[AFG$^+$10] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 209–236. Springer, August 2010.

[AGOT14a] Masayuki Abe, Jens Groth, Miyako Ohkubo, and Mehdi Tibouchi. Structure-preserving signatures from type II pairings. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 390–407. Springer, August 2014.

[AGOT14b] Masayuki Abe, Jens Groth, Miyako Ohkubo, and Mehdi Tibouchi. Unified, minimal and selectively randomizable structure-preserving signatures. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 688–712. Springer, February 2014.

[AH80] Kenneth Appel and Wolfgang Haken. Every planar map is four colorable. *Mathematical Solitaires & Games*, page 145, 1980.

[AH14] Jeremy Avigad and John Harrison. Formally verified mathematics. *Commun. ACM*, 57(4):66–75, April 2014.

[AR07] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptology*, 20(3):395, July 2007.

[BB04] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, May 2004.

[BBG05a] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *Advances in Cryptology–EUROCRYPT 2005*, pages 440–456. Springer, 2005.

[BBG05b] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. Cryptology ePrint Archive, Report 2005/015, 2005.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '01, pages 213–229, London, UK, UK, 2001. Springer-Verlag.

[BF03] Dan Boneh and Matthew Franklin. Identity-based encryption from the weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[BFF$^+$14] Gilles Barthe, Edvard Fagerholm, Dario Fiore, John C. Mitchell, Andre Scedrov, and Benedikt Schmidt. Automated analysis of cryptographic assumptions in generic group models. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 95–112. Springer, August 2014.

[BFF$^+$15] Gilles Barthe, Edvard Fagerholm, Dario Fiore, Andre Scedrov, Benedikt Schmidt, and Mehdi Tibouchi. Strongly-optimal structure preserving signatures from type ii pairings: Synthesis and lower bounds. In Jonathan Katz, editor, *Public-Key Cryptography – PKC 2015*, volume 9020 of *LNCS*, pages 355–376. Springer Berlin Heidelberg, 2015.

[BGHZ11] Gilles Barthe, Benjamin Grégoire, Sylvain Heraud, and Santiago Zanella Béguelin. Computer-aided security proofs for the working cryptographer. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 71–90. Springer, August 2011.

[BGOY07a] Alexandra Boldyreva, Craig Gentry, Adam O'Neill, and Dae Hyun Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson, editors, *ACM CCS 07*, pages 276–285. ACM Press, October 2007.

[BGOY07b] Alexandra Boldyreva, Craig Gentry, Adam O'Neill, and Dae Hyun Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. Cryptology ePrint Archive, Report 2007/438, revised 21 Feb 2010, 2007.

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 258–275. Springer, August 2005.

[BGZB09] Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. Formal certification of code-based cryptographic proofs. In *Proceedings of the 36th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '09, pages 90–101, New York, NY, USA, 2009. ACM.

[Bon98] Dan Boneh. The decision diffie-hellman problem. In JoeP. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer Berlin Heidelberg, 1998.

[Boy08] Xavier Boyen. The uber-assumption family (invited talk). In Steven D. Galbraith and Kenneth G. Paterson, editors, *PAIRING 2008*, volume 5209 of *LNCS*, pages 39–56. Springer, September 2008.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, CCS '93, pages 62–73, New York, NY, USA, 1993. ACM.

[BS02] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324:71–90, 2002.

[BSW13] Karyn Benson, Hovav Shacham, and Brent Waters. The k-BDH assumption family: Bilinear map cryptography from progressively weaker assumptions. In Ed Dawson, editor, *CT-RSA 2013*, volume 7779 of *LNCS*, pages 310–325. Springer, February / March 2013.

[CHL$^+$14] Jung Hee Cheon, Kyoohyung Han, Changmin Lee, Hansol Ryu, and Damien Stehle. Cryptanalysis of the multilinear map over the integers. Cryptology ePrint Archive, Report 2014/906, 2014. `http://eprint.iacr.org/`.

[CLT13] Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 476–493. Springer, August 2013.

[CLT14] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Cryptanalysis of two candidate fixes of multilinear maps over the integers. Cryptology ePrint Archive, Report 2014/975, 2014. `http://eprint.iacr.org/`.

[CM14] Sanjit Chatterjee and Alfred Menezes. Type 2 structure-preserving signature schemes revisited. Cryptology ePrint Archive, Report 2014/635, 2014. `http://eprint.iacr.org/`.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.

[DMB08] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.

[EHK$^+$13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 129–147. Springer, August 2013.

[Fre10] David Mandell Freeman. Converting pairing-based cryptosystems from composite-order groups to prime-order groups. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 44–61. Springer, May 2010.

[FT62] Walter Feit and John G Thompson. A solvability criterion for finite groups and some consequences. *Proceedings of the National Academy of Sciences of the United States of America*, 48(6):968, 1962.

[FT63] Walter Feit and John Thompson. Solvability of groups of odd order. *Pacific Journal of Mathematics*, 13(3), 1963.

[Fuc14] Georg Fuchsbauer. Breaking existential unforgeability of a signature scheme from asiacrypt 2014. Cryptology ePrint Archive, Report 2014/892, 2014.

[GAA$^+$13] Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, Francois Garillot, Stephane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Thery. A machine-checked proof of the odd order theorem. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving*, volume 7998 of *Lecture Notes in Computer Science*, pages 163–179. Springer Berlin Heidelberg, 2013.

[Gen09] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009.

[GGH13] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 1–17. Springer, May 2013.

[Gon08] Georges Gonthier. Formal proof–the four-color theorem. *Notices of the AMS*, 55(11):1382–1393, 2008.

[GPS08] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Appl. Math.*, 156(16):3113–3121, September 2008.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. Acm, 2006.

[GT12] Kristian Gjøsteen and Øystein Thuen. Password-based signatures. In *Public Key Infrastructures, Services and Applications*, pages 17–33. Springer, 2012.

[Hal05] Shai Halevi. A plausible approach to computer-aided cryptographic proofs. Cryptology ePrint Archive, Report 2005/181, 2005. `http://eprint.iacr.org/`.

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In *Advances in Cryptology–EUROCRYPT 2002*, pages 466–481. Springer, 2002.

[HS14] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology - ASIACRYPT 2014*, volume 8873 of *Lecture Notes in Computer Science*, pages 491–511. Springer Berlin Heidelberg, 2014.

[HSW13] Susan Hohenberger, Amit Sahai, and Brent Waters. Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 494–512. Springer, August 2013.

[Jou00] Antoine Joux. A one round protocol for tripartite diffie-hellman. In *Proceedings of the 4th International Symposium on Algorithmic Number Theory*, ANTS-IV, pages 385–394, London, UK, UK, 2000. Springer-Verlag.

[JS08] Tibor Jager and Jörg Schwenk. On the equivalence of generic group models. In Joonsang Baek, Feng Bao, Kefei Chen, and Xuejia Lai, editors, *Provable Security*, volume 5324 of *Lecture Notes in Computer Science*, pages 200–209. Springer Berlin Heidelberg, 2008.

[KL07] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography (Chapman & Hall/Crc Cryptography and Network Security Series)*. Chapman & Hall/CRC, 2007.

[KM10] Neal Koblitz and Alfred Menezes. The brave new world of bodacious assumptions in cryptography. *AMS Notices*, 57:357–365, 2010.

[KSW13] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. *Journal of Cryptology*, 26(2):191–224, April 2013.

[LOS$^+$10] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Advances in Cryptology–EUROCRYPT 2010*, pages 62–91. Springer, 2010.

[LRSW99] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, *SAC 1999*, volume 1758 of *LNCS*, pages 184–199. Springer, August 1999.

[Mat70] Ju V Matijasevič. Enumerable sets are diophantine. In *Dokl. Akad. Nauk SSSR*, volume 191, pages 279–282. World Scientific, 1970.

[Mau05] Ueli M. Maurer. Abstract models of computation in cryptography (invited paper). In Nigel P. Smart, editor, *10th IMA International Conference on Cryptography and Coding*, volume 3796 of *LNCS*, pages 1–12. Springer, December 2005.

[Nec94] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55(2):165–172, 1994.

[Pan14] Yanbin Pan. The security of the hanser-slamanig signature scheme revisited. Cryptology ePrint Archive, Report 2014/915, 2014. `http://eprint.iacr.org/`.

[Rob59] Abraham Robinson. Solution of a problem of tarski. *Fundamenta Mathematicae*, 47(2):179–204, 1959.

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in Cryptology*, pages 47–53, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[Sha07] Hovav Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants. Cryptology ePrint Archive, Report 2007/074, 2007. http://eprint.iacr.org/2007/074.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, May 1997.

[Sip06] Michael Sipser. *Introduction to the Theory of Computation*. Thomson Course Technology, 2nd edition, 2006.

[Uni13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. http://homotopytypetheory.org/book, Institute for Advanced Study, 2013.