On the Tightness of Forward-Secure Signature Reductions

Michel Abdalla[1,2], Fabrice Benhamouda[3], and David Pointcheval[1,2]

[1] Département d'informatique de l'ENS, École normale supérieure,
CNRS, PSL Research University, 75005 Paris, France.
{Michel.Abdalla,David.Pointcheval}@ens.fr
[2] INRIA
[3] IBM Research, Yorktown Heights, NY, USA.
Fabrice.Benhamouda@normalesup.org

Abstract

In this paper, we revisit the security of factoring-based signature schemes built via the Fiat-Shamir transform and show that they can admit tighter reductions to certain decisional complexity assumptions such as the quadratic-residuosity, the high-residuosity, and the ϕ-hiding assumptions. We do so by proving that the underlying identification schemes used in these schemes are a particular case of the lossy identification notion recently introduced by Abdalla et al. at Eurocrypt 2012. Next, we show how to extend these results to the forward-security setting based on ideas from the Itkis-Reyzin forward-secure signature scheme. Unlike the original Itkis-Reyzin scheme, our construction can be instantiated under different decisional complexity assumptions and has a much tighter security reduction. Moreover, we also show that the tighter security reductions provided by our proof methodology can result in concrete efficiency gains in practice, both in the standard and forward-security setting, as long as the use of stronger security assumptions is deemed acceptable. Finally, we investigate the design of forward-secure signature schemes whose security reductions are fully tight.

Contents

1 Introduction **1**

2 Preliminaries **3**

3 Lossy Key-Evolving Identification and Signature Schemes **8**
3.1 Lossy Key-Evolving Identification Scheme 9
3.2 Generalized Fiat-Shamir Transform 11

4 Tighter Security Reductions for Guillou-Quisquater-like Schemes **17**
4.1 Guillou-Quisquater Scheme 18
4.2 Variant of the Itkis-Reyzin Scheme 21

5 Analysis of our Variant of the Itkis-Reyzin Scheme **22**
5.1 Computation of the Exponents $e_1, \ldots, e_T$ 22
5.2 Optimizations 23
5.3 Choice of Parameters 24
5.4 Comparison with Existing Schemes 24

6 Generic Factoring-Based Forward-Secure Signature Scheme **26**
6.1 Generic Factoring-Based Forward-Secure Signature Scheme 26
6.2 An Optimization 28
6.3 Instantiations 29

7 Impossibility Results on Tightness **30**
7.1 Intuition 31
7.2 Key-Verifiable Key-Evolving Signature Scheme 31
7.3 Black-Box Non-Rewinding Reductions 32
7.4 Main Theorem 32

8 Multi-User and Tightly Forward-Secure Signature Schemes **33**
8.1 M-SUF-CMA Signature Schemes 33
8.2 From M-SUF-CMA to Forward-Secure Signature Schemes 34

9 Constructions of Tightly Secure M-SUF-CMA Signature Schemes **35**
9.1 Preliminaries 36
9.2 Construction Based on Commitments and Simulation-Extractable NIZKs 38
9.3 Construction Based on One-Time M-SUF-CMA Schemes 40

Acknowledgments **43**

References **44**

A Relations Between Security Notions **48**

B Mathematical Tools **49**

C Proofs of Security Based on the Forking Lemma for Key-Evolving Collision-Intractable Identification Schemes **52**
C.1 Key-Evolving Collision-Intractable Identification Schemes 52
C.2 Generalized Fiat-Shamir Transformation 53
C.3 Security of the Itkis-Reyzin Scheme 55

1 Introduction

A common paradigm for constructing signature schemes is to apply the Fiat-Shamir transform [FS87] to a secure three-move canonical identification protocol. In these protocols, the prover first sends a commitment to the verifier, which in turn chooses a random string from the challenge space and sends it back to the prover. Upon receiving the challenge, the prover sends a response to the verifier, which decides whether or not to accept based on the conversation transcript and the public key. To obtain the corresponding signature scheme, one simply makes the signing and verification algorithms non-interactive by computing the challenge as the hash of the message and the commitment. As shown by Abdalla et al. in [AABN02], the resulting signature scheme can be proven secure in the random oracle model as long as the identification scheme is secure against passive adversaries and the commitment has large enough min-entropy. Unfortunately, the reduction to the security of the identification scheme is not tight and loses a factor q_h, where q_h denotes the number of queries to the random oracle.

If one assumes additional properties about the identification scheme, one can avoid impossibility results such as those in [GBL08, PV05, Seu12] and obtain a signature scheme with a tighter proof of security. For instance, in [MR02], Micali and Reyzin introduced a new method for converting identification schemes into signature schemes, known as the "swap method," in which they reverse the roles of the commitment and challenge. More precisely, in their transform, the challenge is chosen uniformly at random from the challenge space and the commitment is computed as the hash of the message and the challenge. Although they only provided a tight security proof for the modified version of Micali's signature scheme [Mic94], their method generalizes to any scheme in which the prover can compute the response given only the challenge and the commitment, such as the factoring-based schemes in [FFS88, FS87, GQ88, OO90, OS91][1]. This is due to the fact that the prover in these schemes possesses a trapdoor (such as the factorization of the modulus in the public key) which allows it to compute the response. On the other hand, their method does not apply to discrete-log-based identification schemes in which the prover needs to know the discrete log of the commitment when computing the response, such as in [Sch90].

In 2003, Katz and Wang [KW03] showed that tighter security reductions can be obtained even with respect to the Fiat-Shamir transform, by relying on a proof of membership rather than a proof of knowledge. In particular, using this idea, they proposed a signature scheme with a tight security reduction to the hardness of the DDH problem. They also informally mentioned that one could obtain similar results based on the quadratic-residuosity problem by relying on a proof that shows that a set of elements in $\mathbb{Z}_N^*$ are all quadratic residues. This result was recently extended to other settings by Abdalla et al. [AFLT12], who presented three new signature schemes based on the hardness of the short exponent discrete log problem [PS98, vW96], on the worst-case hardness of the shortest vector problem in ideal lattices [LM06, PR06], and on the hardness of the Subset Sum problem [IN96, MM11]. Additionally, they also formalized the intuition in [KW03] by introducing the notion of lossy identification schemes and showing that any such scheme can be transformed into a signature scheme via the Fiat-Shamir transform while preserving the tightness of the reduction.

Tight security from lossy identification. In light of these recent results, we revisit in this paper the security of factoring-based signature schemes built via the Fiat-Shamir transform. Even though the swap method from [MR02] could be applied in this setting (resulting in a slightly different scheme), our first contribution is to show that these signature schemes already admit tight security reductions to certain decisional complexity assumptions such as the quadratic-residuosity, the high-residuosity [Pai99], and the ϕ-hiding [CMS99] assumptions. We do so by showing that the underlying identification schemes used in these schemes are a particular case of a lossy identification scheme [AFLT12]. As shown in Section 4.1 in the case of the Guillou-Quisquater signature scheme [GQ88], our tighter security reduction can result in concrete efficiency gains with respect to the swap method. However, this comes at the cost of relying on a stronger security assumption, namely the ϕ-hiding [CMS99] assumption, instead of the plain RSA

[1] In [BPS16], Bellare, Poettering, and Stebila call these schemes *trapdoor* and provide a formal definition for it.

assumption. Nevertheless, as explained by Kakvi and Kiltz in [KK12], for carefully chosen parameters, the currently best known attack against the ϕ-hiding problems consists in factorizing the corresponding modulus, which is also the best known attack against the plain RSA assumption.

More generally, one needs to be careful when comparing the tightness of different security reductions, especially if the underlying complexity assumptions and security models are different. In order to have meaningful comparisons, we would like to stress that we focus mainly on schemes whose security holds in the random oracle model [BR93] and whose underlying computational assumptions have comparable complexity estimates, as in the case of the ϕ-hiding [CMS99] and plain RSA assumptions for carefully chosen parameters.

Tighter reductions for forward-secure signatures. Unlike the swap method of Micali and Reyzin, the prover in factoring-based signature schemes built via the Fiat-Shamir transform does not need to know the factorization of the modulus in order to be able to compute the response. Using this crucial fact, the second main contribution of this paper is to extend our results to the forward-security setting. To achieve this goal, we first introduce in Section 3 the notion of lossy key-evolving identification schemes and show how the latter can be turned into forward-secure signature schemes using a generalized version of the Fiat-Shamir transform. As in the case of standard signature schemes, this transformation does not incur a loss of a factor q_h in the security reduction. Nevertheless, we remark that the reduction is not entirely tight as we lose a factor T corresponding to the total number of time periods.

After introducing the notion of lossy key-evolving identification schemes, we show in Section 4.2 that a variant of the Itkis-Reyzin forward-secure signature scheme [IR01] (which can be seen as an extension of the Guillou-Quisquater scheme to the forward-security setting) admits a much tighter security reduction, albeit to a stronger assumption than the plain RSA assumption, namely the ϕ-hiding assumption. However, we point out that the most efficient variant of the Itkis-Reyzin scheme does not rely on the plain RSA assumption but on the strong RSA assumption. There is currently no known reduction between the strong RSA and the ϕ-hiding assumption.

Concrete security. As in the case of standard signature schemes, the tighter security reductions provided by our proof methodology can result in concrete efficiency gains in practice. More specifically, as we show in Section 5, our variant of the Itkis-Reyzin scheme outperforms the original scheme for most concrete choices of parameters.

Generic factoring-based signatures and forward-secure signatures. As an additional contribution, we show in Section 6 that all the above-mentioned schemes can be seen as straightforward instantiations of a generic factoring-based forward-secure signature scheme. This enables us to not only easily prove the security properties of these schemes, but to also design a new forward-secure scheme based on a new assumption, the gap 2^t-residuosity.[2] This assumption has been independently considered and proven secure by Benhamouda, Herranz, Joye, and Libert in [JL13, BHJL16], under a variant of the quadratic residuosity assumption together with a new reasonable assumption called the "squared Jacobi symbol" assumption.

Impossibility and existential results for tight forward-secure signature schemes. As pointed out above, the reductions for our forward-secure signature schemes are not entirely tight as we still lose a factor T corresponding to the total number of time periods. Hence, an interesting question to ask is whether it is possible to provide a better security reduction for these schemes. To answer this question, we first show in Section 7 that the loss of a factor T in the proof of forward security cannot be avoided for a large class of key-evolving signature schemes, which includes the ones considered so far. This is achieved by extending Coron's impossibility result in [Cor02] to the forward-secure setting.

Next, in Sections 8 and 9, we show how to avoid these impossibility results and build forward-secure signature schemes whose security reductions are fully tight. To do that, we first propose a new notion of

[2] We originally called this assumption the strong-2^t-residuosity in [ABP13]. In this full version, we prefer to use the same name as in [JL13].

security for signature schemes in Section 8: strong unforgeability in a multi-user setting with corruptions (M-SUF-CMA). This notion is related to the security definition given by Menezes and Smart in [MS04] but unlike theirs, our notion takes into account user corruptions. Next, we propose generic transformations from M-SUF-CMA signature schemes to forward-secure signature schemes which preserve tightness. Finally, in Section 9, we provide several instantiations of M-SUF-CMA signature schemes with tight security reductions to standard non-interactive hard problems. The results in Sections 8 and 9 are mostly of theoretical interest as the schemes that we obtain are significantly less efficient than the ones in preceding sections.

In an independent paper [BJLS16, Section 5.1], Bader et al. also studied signature schemes in a multi-user setting (with corruptions). Using a meta-reduction, they showed that M-SUF-CMA cannot be tightly reduced to standard non-interactive hard problems, if secret keys can be re-randomized. On the one hand, contrary to our meta-reduction for forward-secure signature schemes, their meta-reduction allows for rewinding. On the other hand, if we forget about rewindings, our impossibility result implies the one in [BJLS16] for M-SUF-CMA signature schemes, as forward-secure signature schemes can be constructed from M-SUF-CMA signature schemes with a tight reduction.[3]

Publication Note. An abridged version of this paper appeared in the proceedings of PKC 2013 [ABP13]. In this version, we give more precise and formal security definitions and statements, we include complete proofs of security, and we provide *new impossibility and existential results* for tight forward-secure signature schemes. Most notably, we demonstrate that the loss of a factor T corresponding to the total number of time periods cannot be avoided in the proof of forward security for a large class of key-evolving signature schemes, including all the schemes considered in [ABP13]. In addition, we also show how to avoid these impossibility results and build forward-secure signature schemes whose security reductions are fully tight.

Organization. Section 2 recalls some basic definitions and complexity assumptions used in the paper. Section 3 introduces lossy key-evolving identification schemes and shows how to transform them into forward-secure signature schemes. Section 4 applies our security proof methodology to two cases: the Guillou-Quisquater signature scheme and its extension to the forward-secure setting, which is a variant of the Itkis-Reyzin forward-secure signature scheme in [IR01]. Section 5 compares our variant of the Itkis-Reyzin forward-secure signature scheme to the original one and to the MMM scheme by Malkin, Micciancio, and Miner [MMM02]. Section 6 introduces a generic factoring-based forward-secure signature scheme along with various instantiations. Section 7 provides further results regarding the reduction tightness of forward-secure signature schemes. In particular, it shows that the loss of a factor T in the proof of forward security cannot be avoided for a large class of key-evolving signature schemes, which includes the ones considered in the previous sections. Sections 8 and 9 show how to avoid the impossibility results in Section 7 and build forward-secure signature schemes with tight security reductions. The appendix provides additional results regarding forward-secure signature schemes. More precisely, Appendix A presents a few relations between different security notions for forward-signature schemes. Appendix B presents several results used in the security analysis of our signature schemes. Finally, Appendix C provides additional proofs used in the concrete security analysis in Section 5.

2 Preliminaries

2.1 Notation and Conventions

Let $\mathbb{N}$ denote the set of natural numbers. If $N \in \mathbb{N}$ and $N \geq 2$, then $\mathbb{Z}_N = \mathbb{Z}/N\mathbb{Z}$ is the ring of integers modulo N, and $\mathbb{Z}_N^*$ is its group of units. If $e, N \in \mathbb{N}$ and $e, N \geq 2$, then an element $y \in \mathbb{Z}_N^*$ is an e-residue modulo N if there exists an element $x \in \mathbb{Z}_N^*$ such that $y = x^e \bmod N$. We denote the set of e-residues modulo N by $\mathsf{HR}_N[e]$.

[3] Furthermore, M-SUF-CMA signature schemes with re-randomizable secret keys yield forward-secure signature schemes in the class we consider in our impossiblity result, if the secret key is re-randomized before signing.

If $n \in \mathbb{N}$, then $\{0,1\}^n$ denotes the set of n-bit strings, and $\{0,1\}^*$ is the set of all bit strings. The empty string is denoted $\perp$. An empty table T is denoted [], and $\mathsf{T}[x]$ is the value of the table at index x and is equal to $\perp$ if undefined. If x is a string then $|x|$ denotes its length, and if S is a set then $|S|$ denotes its size. If S is finite, then $x \xleftarrow{\$} S$ denotes the assignment to x of an element chosen uniformly at random from S. If $\mathcal{A}$ is an algorithm, then $y \leftarrow \mathcal{A}(x)$ denotes the assignment to y of the output of $\mathcal{A}$ on input x, and if $\mathcal{A}$ is randomized, then $y \xleftarrow{\$} \mathcal{A}(x)$ denotes that the output of an execution of $\mathcal{A}(x)$ (with fresh coins) is assigned to y. Unless otherwise indicated, an algorithm may be randomized. We denote by $k \in \mathbb{N}$ the security parameter. Let $\mathbb{P}$ denote the set of primes and $\mathbb{P}_{\ell_e}$ denote the set of primes of length ℓ_e. Most of our schemes are in the random oracle model [BR93].

2.2 Games

The definitions and proofs in this paper use code-based game-playing techniques [BR06]. In such games, there exist procedures for initialization (**Initialize**) and finalization (**Finalize**) and procedures to respond to adversary oracle queries. A game G is executed with an adversary $\mathcal{A}$ as follows. First, **Initialize** executes and its outputs are the inputs to $\mathcal{A}$. Then $\mathcal{A}$ executes, its oracle queries being answered by the corresponding procedures of G. When $\mathcal{A}$ terminates, its output becomes the input to the **Finalize** procedure. The output of the latter, denoted $\mathrm{G}(\mathcal{A})$, is called the output of the game, and "$\mathrm{G}(\mathcal{A}) \Rightarrow y$" denotes the event that the output takes a value y. The running time of an adversary is the worst case time of the execution of the adversary with the game defining its security, so that the execution time of the called game procedures is included.

Review of code-based game-playing proofs. We recall some background on code-based game-playing. The boolean flag bad is assumed initialized to false. We say that games $\mathrm{G}_i, \mathrm{G}_j$ are identical until bad if their programs differ only in statements that (syntactically) follow the setting of bad to true. For examples, games $\mathrm{G}_0, \mathrm{G}_1$ of Figure 3.2 are identical until bad. Let us now recall two lemmas stated in [BR06] and in [BNN07].

Lemma 2.1 ([BR06]) *Let $\mathrm{G}_i, \mathrm{G}_j$ be identical-until-bad games, and $\mathcal{A}$ an adversary. Then we have $|\Pr[\mathrm{G}_i(\mathcal{A}) \Rightarrow 1] - \Pr[\mathrm{G}_j(\mathcal{A}) \Rightarrow 1]| \leq \Pr[\mathrm{G}_i(\mathcal{A}) \text{ sets } \mathsf{bad}]$.*

Lemma 2.2 ([BNN07]) *Let $\mathrm{G}_i, \mathrm{G}_j$ be identical-until-bad games, and $\mathcal{A}$ an adversary. Let $\mathsf{Good}_i, \mathsf{Good}_j$ be the events that bad is never set in games $\mathrm{G}_i, \mathrm{G}_j$, respectively. Then, $\Pr[\mathrm{G}_i(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_i] = \Pr[\mathrm{G}_j(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_j]$.*

2.3 Statistical Distance

Let D_1 and D_2 be two probability distributions over a finite set $\mathcal{S}$ and let X and Y be two random variables with these two respective distributions. The statistical distance between D_1 and D_2 is also the statistical distance between X and Y:

$$\frac{1}{2} \sum_{x \in \mathcal{S}} |\Pr[X = x] - \Pr[Y = x]| .$$

If the statistical distance between D_1 and D_2 is less than or equal to ε, we say that D_1 and D_2 are ε-close or are ε-statistically indistinguishable. If the D_1 and D_2 are 0-close, we say that D_1 and D_2 are perfectly indistinguishable.

We use the following lemma.

Lemma 2.3 *Let S_0 and S_1 two finite sets such that $S_1 \subseteq S_0$. Let D_0 and D_1 be the uniform distributions over S_0 and S_1 respectively. Let $N_0 = |S_0|$ and $N_1 = |S_1|$ be the cardinals of S_0 and S_1 respectively. Then, the statistical distance between D_0 and D_1 is $1 - N_1/N_0$.*

Proof: The statistical distance is:

$$D = \frac{1}{2} \cdot \left(\sum_{x \in S_1} \left| \Pr_{y \xleftarrow{\$} \mathsf{D}_0}[y = x] - \Pr_{y \xleftarrow{\$} \mathsf{D}_1}[y = x] \right| + \sum_{x \in S_0 \setminus S_1} \left| \Pr_{y \xleftarrow{\$} \mathsf{D}_0}[y = x] - \Pr_{y \xleftarrow{\$} \mathsf{D}_1}[y = x] \right| \right)$$
$$= \frac{1}{2} \cdot \left(N_1 \cdot \left| \frac{1}{N_0} - \frac{1}{N_1} \right| + (N_0 - N_1) \cdot \left| \frac{1}{N_0} - 1 \right| \right) = \frac{1}{2} \cdot \left(1 - \frac{N_1}{N_0} + 1 - \frac{N_1}{N_0} \right) = 1 - \frac{N_1}{N_0}.$$

∎

2.4 Complexity Assumptions

The security of the signature schemes being analyzed in this paper will be based on decisional assumptions over composite-order groups: the quadratic residuosity assumption, the high residuosity assumption, the ϕ-hiding assumption, and a new assumption called the gap 2^t-residuosity. We also need to recall the strong RSA assumption to be able to compare our scheme with the Itkis-Reyzin scheme [IR01].

For all these assumptions, the underlying problem consists in distinguishing two distributions D_1 and D_2. More precisely, an adversary $\mathcal{D}$ is said to (t, ε)-solve or (t, ε)-break the underlying problem if it runs in time t and

$$\left| \Pr\left[\mathcal{D}(x) = 1 \;\middle|\; x \xleftarrow{\$} D_1 \right] - \Pr\left[\mathcal{D}(x) = 1 \;\middle|\; x \xleftarrow{\$} D_2 \right] \right| \leq \varepsilon.$$

Then the underlying problem is said to be (t, ε)-hard if no adversary can (t, ε)-solve it.

ϕ-HIDING ASSUMPTION [CMS99, KOS10]. The **ϕ-hiding** assumption, introduced by Cachin, Micali, and Stadler in [CMS99], states that it is hard for an adversary to tell whether a prime number e divides the order $\phi(N)$ of the group $\mathbb{Z}_N^*$. In this paper, we use a very slight variant of the formulation in [KOS10].[4]

More formally, let ℓ_N be a function of k, let ℓ_e be a public positive constant smaller than $\frac{1}{4}\ell_N$. Let RSA_{ℓ_N} denote the set of all tuples (N, p_1, p_2) such that $N = p_1 p_2$ is ℓ_N-bit number which is the product of two distinct $\ell_N/2$-bit primes, as in [KOS10]. N is called an RSA modulus. Likewise, let R be a relation on p_1 and p_2. We denote by $\mathsf{RSA}_{\ell_N}[R]$ the subset of RSA_{ℓ_N} for which the relation R holds. The ϕ-hiding assumption states that the two following distributions are computationally indistinguishable:

$$\{(N, e) \mid e \xleftarrow{\$} \mathbb{P}_{\ell_e},\ (N, p_1, p_2) \xleftarrow{\$} \mathsf{RSA}_{\ell_N}[\gcd(e, \phi(N)) = 1]\}$$
$$\{(N, e) \mid e \xleftarrow{\$} \mathbb{P}_{\ell_e},\ (N, p_1, p_2) \xleftarrow{\$} \mathsf{RSA}_{\ell_N}[p_1 = 1 \bmod e]\},$$

where $\phi(N)$ is the order of $\mathbb{Z}_N^*$.

We remark that the these two distributions can be sampled efficiently if we assume the widely-accepted Extended Riemann Hypothesis (Conjecture 8.4.4 of [BS96]).

QUADRATIC RESIDUOSITY. The **quadratic-residuosity** assumption states that it is hard to distinguish a 2-residue (a.k.a, a quadratic residue) from an element of Jacobi symbol 1, modulo an RSA modulus N.

More formally, let N be an RSA modulus. We recall that $\mathsf{HR}_N[2]$ denotes the set of all 2-residues modulo N. Let $\mathsf{J}_N[2]$ be the set of elements in $\mathbb{Z}_N^*$ with Jacobi symbol 1. The quadratic-residuosity assumption states that the two following distributions are computationally indistinguishable:

$$\{(N, y) \mid N \xleftarrow{\$} \mathsf{RSA}_{\ell_N},\ y \xleftarrow{\$} \mathsf{HR}_N[2]\}$$
$$\{(N, y) \mid N \xleftarrow{\$} \mathsf{RSA}_{\ell_N},\ y \xleftarrow{\$} \mathsf{J}_N[2] \setminus \mathsf{HR}_N[2]\}.$$

HIGH RESIDUOSITY. Let e be an RSA modulus and $N = e^2$. The **high-residuosity** assumption states that it is hard to distinguish a e-residue modulo N from an element from $\mathbb{Z}_N^*$.

[4] We enforce that e is co-prime to $\phi(N)$, while in [KOS10], they enforce instead that $\phi(N)$ is divisible by another randomly chosen ℓ_e-bit prime e'. This slightly simplifies proofs, notation, and bounds, but the original assumption could be used as well.

More formally, let $\mathsf{J}_N[e] = \mathbb{Z}_N^*$. The high-residuosity assumption states that the two following distributions are computationally indistinguishable:

$$\{(N, e, y) \mid e \xleftarrow{\$} \mathsf{RSA}_{\ell_N},\ N \leftarrow e^2,\ y \xleftarrow{\$} \mathsf{HR}_N[e]\}$$
$$\{(N, e, y) \mid e \xleftarrow{\$} \mathsf{RSA}_{\ell_N},\ N \leftarrow e^2,\ y \xleftarrow{\$} \mathsf{J}_N[e] \setminus \mathsf{HR}_N[e]\}.$$

GAP 2^t-RESIDUOSITY. We introduce the **gap 2^t-residuosity** assumption, that states that it is hard for an adversary to decide whether a given element y (in $\mathbb{Z}_N^*$) of Jacobi symbol 1 is a 2^t-residue or is even not a 2-residue, when 2^t divides $p_1 - 1$ and $p_2 - 1$.

More formally, this assumption states that the two following distributions are computationally indistinguishable:

$$\{(N, e, y) \mid N \xleftarrow{\$} \mathsf{RSA}_{\ell_N}[p_1, p_2 = 1 \bmod 2^t],\ y \xleftarrow{\$} \mathsf{HR}_N[2^t]\}$$
$$\{(N, e, y) \mid N \xleftarrow{\$} \mathsf{RSA}_{\ell_N}[p_1, p_2 = 1 \bmod 2^t],\ y \xleftarrow{\$} \mathsf{J}_N[2] \setminus \mathsf{HR}_N[2]\}.$$

This assumption has been independently considered and proven secure by Benhamouda, Herranz, Joye, and Libert in [JL13, BHJL16], under a variant of the quadratic residuosity assumption together with a new reasonable assumption called the "squared Jacobi symbol" assumption.

STRONG RSA. The **strong RSA** assumption states that, given an element $y \in \mathbb{Z}_N^*$, it is hard for an adversary to find an integer $2 \leq e \leq 2^{\ell_e}$ and an element $x \in \mathbb{Z}_N^*$ such that $y = x^e \bmod N$, where ℓ_e is a function of the security parameter k. In this article, we actually use the variant of the strong RSA assumption described in [IR01]. As explained in the latter article, compared to the original version of the assumption introduced independently in [BP97] and in [FO97], we restrict N to be a product of two safe primes[5] and we restrict e to be at most 2^{ℓ_e} for some value ℓ_e. We remark that, formally, we have defined a family of assumptions indexed by ℓ_e, a function of k.

2.5 Forward-Secure Signature Schemes

A forward-secure signature scheme is a key-evolving signature scheme in which the secret key is updated periodically while the public key remains the same throughout the lifetime of the scheme [BM99]. Each time period has a secret signing key associated with it, which can be used to sign messages with respect to that time period. The validity of these signatures can be checked with the help of a verification algorithm. At the end of each time period, the signer in possession of the current secret key can generate the secret key for the next time period via an update algorithm. Moreover, old secret keys are erased after a key update.

Formally, a key-evolving signature scheme is defined by a tuple of algorithms $\mathcal{FS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ and a message space $\mathcal{M}$, providing the following functionality. Via $(pk, sk) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$, a user can run the probabilistic key generation algorithm KG to obtain a pair (pk, sk_1) of public and secret keys for a given security parameter k and a given total number of periods T. sk_1 is the secret key associated with time period 1. Via $sk_{i+1} \leftarrow \mathsf{Update}(sk_i)$, the user in possession of the secret key sk_i associated with time period $i \leq T$ can generate a secret key sk_{i+1} associated with time period $i + 1$. By convention, $sk_{T+1} = \perp$. Via $(\sigma, i) \xleftarrow{\$} \mathsf{Sign}(sk_i, M)$, the user in possession of the secret key sk_i associated with time period $i \leq T$ can generate a signature (σ, i) for a message $M \in \mathcal{M}$ for period i. Finally, via $d \leftarrow \mathsf{Ver}(pk, (\sigma, i), M)$, one can run the deterministic verification algorithm to check if σ is a valid signature for a message $M \in \mathcal{M}$ for period i and public key pk, where $d = 1$ if the signature is correct and 0 otherwise. For correctness, it is required that for all honestly generated keys $(sk_1, \ldots, sk_T)$ and for all messages $M \in \mathcal{M}$, $\mathsf{Ver}(pk, \mathsf{Sign}(sk_i, M), M) = 1$ holds with all but negligible probability.

[5] A prime number p is safe if it can be written as $p = 2q + 1$, where q is a prime number.

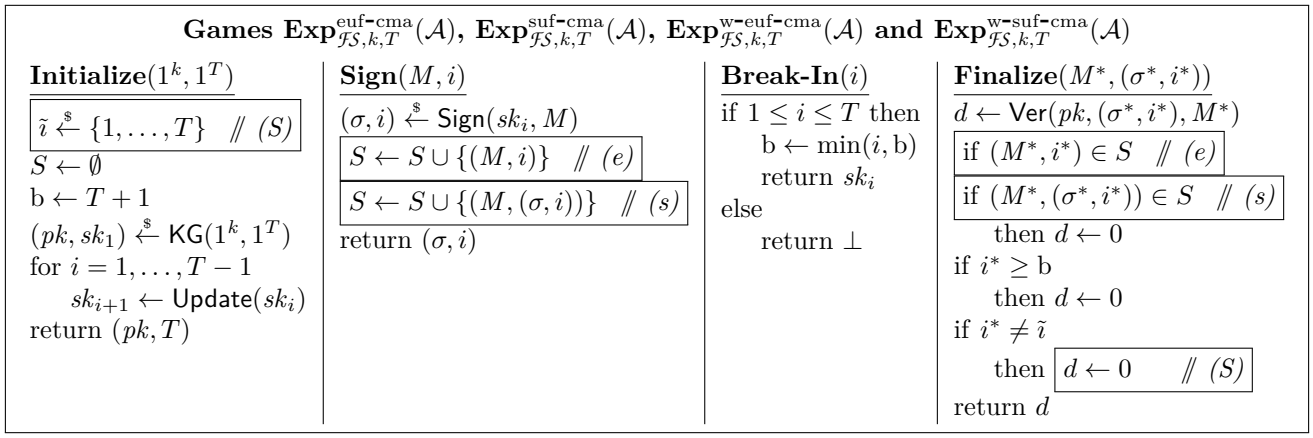

Figure 2.1: Games defining the EUF-CMA, SUF-CMA, W-EUF-CMA and W-SUF-CMA security of a key-evolving signature scheme $\mathcal{FS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$. Boxed lines marked *(e)* are only for the existential variants, boxed lines marked *(s)* are only for the strong variants, and boxed lines marked *(S)* are only for the selective variants.

Existential and strong forward security. Informally, a key-evolving signature scheme is **existentially forward-secure** under adaptive chosen-message attack (EUF-CMA), if it is infeasible for an adversary —also called forger— to forge a signature σ^* on a message M^* for a time period i^*, even with access to the secret key for a period $i > i^*$ (and thus to all the subsequent secret keys; this period i is called the break-in period) and to sign messages of its choice for any period (via a signing oracle), as long as he has not requested a signature on M^* for period i^* to the signing oracle.

This notion is a generalization of the existential unforgeability under adaptive chosen-message attacks (EUF-CMA for signature schemes) [GMR84] to key-evolving signature scheme. It is a slightly stronger variant of the definition in [BM99]. Compared to [BM99], we do not restrict the adversary to only perform signing queries with respect to the current time period and we allow multiple **Break-In** queries (only the break-in period taken into account is the minimum of all these periods). The advantage of the first change is that the game is simpler than the one with the definition of Bellare and Miner in [BM99]. Concerning the second change, the classical notion which only allows a single **Break-In** query is equivalent, as it is possible to guess which query corresponds to the minimum period. However, that does not preserve the tightness of the reduction. Anyway, it seems that most of the current schemes (maybe even all of them) also satisfy our stronger definition, using nearly the same reductions.

In the remainder of the paper, we also use a stronger notion: **(strong) forward security** (SUF-CMA). In this notion, the forger is allowed to produce a signature σ^* on a message M^* for a period i^*, such that the triple (M^*, i^*, σ^*) is different from all the triples produced by the signing oracle.

More formally, let us consider the games $\mathbf{Exp}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ and $\mathbf{Exp}^{\text{suf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ depicted in Figure 2.1. We then say that $\mathcal{FS}$ is $(t, q_h, q_s, \varepsilon)$-existentially-forward-secure, if for any adversary $\mathcal{A}$ running in time at most t and making at most q_h queries to the random oracle and q_s queries to the signing oracle: $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) = \Pr\left[\mathbf{Exp}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) \Rightarrow 1\right] \le \varepsilon$. And $\mathcal{FS}$ is $(t, q_h, q_s, \varepsilon)$-forward-secure, if for any such adversary: $\mathbf{Adv}^{\text{suf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) = \Pr\left[\mathbf{Exp}^{\text{suf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) \Rightarrow 1\right] \le \varepsilon$.

Selective security notions. In addition to the previous classical security notions, we also consider two weaker notions: **selective forward security** and **selective existential forward security**, which are very useful to compare different schemes. In these notions, the time period of the forgery is chosen at the beginning at random but not disclosed to the adversary, and the adversary loses when it does not produce a forgery for the chosen time period. We could have opted for a more classical selective version where the adversary chooses the time period of the forgery at the beginning, but that would have made notation more cumbersome.

More precisely, when defining the selective forward security and selective existential forward security

notions, the challenger of the adversary, picks a period $\tilde{i}$ uniformly at random at the beginning and reject the forged signature if it does not correspond to the period $\tilde{i}$, as in the games $\mathbf{Exp}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ and $\mathbf{Exp}^{\text{w-suf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ depicted in Figure 2.1. Then we say that a key-evolving signature scheme is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-(existentially)-forward-secure if there is no adversary $\mathcal{A}$ (running in time at most t, doing at most q_h requests to the random oracle, and q_s requests of signatures), such that, with probability at least δ, the challenger chooses a period $\tilde{i}$ and a key pair (pk, sk_1), such that $\mathcal{A}$ forges a correct signature for period $\tilde{i}$ with probability at least ε. The idea of this definition is to separate the success probability for a given period and a given key pair, from the choice of the period and the key pair. This enables us to repeat the experiments with the same period and key pair to increase the success probability of the adversary (for a given period and key pair). The main reason we need to keep the same period and key pair at each repetition is that our reduction "embeds" the challenge of the underlying assumption into them and this challenge cannot be changed between two repetitions, as the assumptions we use are not known to be random self-reducible.

Finally, we remark that our selective notions are actually extensions of the security definition of Micali and Reyzin in [MR02]. These notions are weaker than the previous ones in the following way: if a scheme is $(t, q_h, q_s, T\varepsilon\delta)$-(existentially)-forward-secure, then it is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-(existentially)-forward-secure, as proven in Appendix A.

Remark 2.4 In order to be able to compare different schemes, as in [MR02], we suppose that any attacker which $(t, q_h, q_s, \varepsilon)$-breaks the selective forward security (where there is no δ and ϵ is the classical success probability here) of a scheme also $(t, q_h, q_s, \varepsilon, \delta = 1/2)$-breaks it.[6] In other words, $(t, q_h, q_s, \varepsilon, 1/2)$-selective forward security implies $(t, q_h, q_s, T\varepsilon)$-forward security. As shown in Section 5.4, it enables us to do a quite fair comparison, if we consider that a $(t, q_h, q_s, T\varepsilon)$-forward-secure scheme has $\log_2(t/(T\varepsilon))$ bits of security, which is the intuitive notion of security.

At first, we might think that just assuming that a $(t, q_h, q_s, T\varepsilon)$-forward-secure scheme provides $\log_2(t/(T\varepsilon))$ bits of security should be sufficient to do fair comparisons. This would indeed be sufficient for our new security reductions, as they basically ensure that if some assumption (e.g., the ϕ-hiding assumption) is (t', ε')-hard, then the signature scheme is $(t, q_h, q_s, T\varepsilon)$-forward-secure for $t \approx t'$ and $\varepsilon \approx \varepsilon'$. But unfortunately, security reductions based on the forking lemma (to which we want to compare our new security reductions) only ensure that if some assumption (e.g., the RSA assumption) is (t', ε')-hard, then the signature scheme is $(t, q_h, q_s, T\varepsilon)$-forward-secure for $t \approx t'$ and $\varepsilon \approx \varepsilon'^2/q_h$. In that case, $\log_2(t/(T\varepsilon))$ is even not well-defined. That is why, following [MR02], we introduced the notions of $(t, q_h, q_s, \varepsilon, \delta)$-selective-(existential)-forward-security to solve this problem (see Theorem C.1 and also the discussions around Theorem 2 in [MR02]).

More details on the relations between these security notions can be found in Appendix A.

3 Lossy Key-Evolving Identification and Signature Schemes

In this section, we present a new notion, called lossy key-evolving identification scheme, which combines the notions of lossy identification schemes [AFLT12], which can be transformed to tightly secure signature scheme, and key-evolving identification schemes [BM99], which can be transformed to forward-secure signature via a generalized Fiat-Shamir transform (not necessarily tight, and under some conditions). Although this new primitive is not very useful for practical real-world applications, it is a tool that will enable us to construct forward-secure signatures with tight reductions, via the generalized Fiat-Shamir transform described in Section 3.2.

[6] 1/2 is just an arbitrary constant. It can be any reasonable constant.

3.1 Lossy Key-Evolving Identification Scheme

The operation of a key-evolving identification scheme is divided into time periods $1, \ldots, T$, where a different secret is used in each time period, and such that the secret key for a period $i+1$ can be computed from the secret key for the period i. The public key remains the same in every time period. In this paper, a key-evolving identification scheme is a three-move protocol in which the prover first sends a **commitment** cmt to the verifier, then the verifier sends a **challenge** ch uniformly at random, and finally the prover answers by a **response** rsp. The verifier's final decision is a deterministic function of the conversation with the prover (the triple (cmt, ch, rsp)), of the public key, and of the index of the current time period.

Informally, a lossy key-evolving identification scheme has $T+1$ types of public keys: normal public keys, which are used in the real protocol, and i-lossy public keys, for $i \in \{1, \ldots, T\}$, which are such that no prover (even not computationally bounded) should be able to make the verifier accept for the period i with non-negligible probability. Furthermore, for each period i, it is possible to generate an i-lossy public key, such that the latter is indistinguishable from a normal public key even if the adversary is given access to any secret key for period $i' > i$.

More formally, a lossy key-evolving identification scheme $\mathcal{ID}$ is defined by a tuple $(\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \mathcal{C}, \mathsf{Ver})$ such that:

- KG is the normal key generation algorithm which takes as input the security parameter k and the number of periods T and outputs a pair (pk, sk_1) containing the public key and the prover's secret key for the first period.
- LKG is the lossy key generation algorithm which takes as input the security parameter k and the number of periods T and a period i and outputs a pair (pk, sk_{i+1}) containing an i-lossy public key pk and a prover's secret key for period $i+1$ ($sk_{T+1} = \perp$).
- Update is the deterministic secret key update algorithm which takes as input a secret key sk_i for period i and outputs a secret key sk_{i+1} for period $i+1$ if sk_i is a secret key for some period $i < T$, and $\perp$ otherwise. We write Update^j the function Update composed j times with itself ($\mathsf{Update}^j(sk_i)$ is a secret key sk_{i+j} for period $i+j$, if $i+j \leq T$).
- Prove is the prover algorithm which takes as input the secret key for the current period, the current conversation transcript (and the current state st associated with it, if needed) and outputs the next message to be sent to the verifier, and the next state (if needed). We suppose that any secret key sk_i for period i always contains i, and so i is not an input of Prove.
- $\mathcal{C}$ is the set of possible challenges that can be sent by the verifier. The set $\mathcal{C}$ might implicitly depend on the public key. In our constructions, it is of the form $\{0, \ldots, \mathfrak{c}-1\}^\ell$.
- Ver is the deterministic verification algorithm which takes as input the conversation transcript and the period i and outputs 1 to indicate acceptance, and 0 otherwise.

A randomized transcript generation oracle $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$ is associated to each $\mathcal{ID}$, k, and (pk, sk_i). $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$ takes no inputs and returns a random transcript of an "honest" execution for period i. More precisely, the transcript generation oracle $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$ is defined as follows:

function $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$
  $(cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)$; $ch \xleftarrow{\$} \mathcal{C}$; $rsp \xleftarrow{\$} \mathsf{Prove}(sk_i, cmt, ch, st)$
  return (cmt, ch, rsp)

An identification scheme is said to be lossy if it has the following properties:

1. **Completeness of normal keys.** $\mathcal{ID}$ is said to be complete, if for every period i, every security parameter k and all honestly generated keys $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k)$, $\mathsf{Ver}(pk, cmt, ch, rsp, i) = 1$ holds with probability 1 when $(cmt, ch, rsp) \xleftarrow{\$} \mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}()$, with $sk_i = \mathsf{Update}^{i-1}(sk_1)$.

2. **Simulatability of transcripts.** Let (pk, sk_1) be the output of $\mathsf{KG}(1^k)$ for a security parameter k, and sk_i be the output of $\mathsf{Update}^{i-1}(sk_1)$. Then, ID is said to be ε-simulatable if there exists a probabilistic polynomial time simulator $\widetilde{\mathsf{Tr}}^{\mathit{ID}}_{pk,i,k}$ with no access to any secret key, which can generate transcripts $\{(cmt, ch, rsp)\}$ whose distribution is statistically indistinguishable from the transcripts output by $\mathsf{Tr}^{\mathit{ID}}_{pk,sk_i,k}$, where ε is an upper-bound for the statistical distance. When $\varepsilon = 0$, then ID is said to be perfectly simulatable.

   This property is also often called statistical honest-verifier zero-knowledge [GMR89, BMO90, Cra96].

3. **Key indistinguishability.** Consider the experiments $\mathbf{Exp}^{\text{ind-keys-real}}_{\mathit{ID},k}(\mathcal{D})$ and $\mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathit{ID},k}(\mathcal{D})$, defined as follows:

$$\begin{array}{l|l}
\mathbf{Exp}^{\text{ind-keys-real}}_{\mathit{ID},k}(\mathcal{D}) & \mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathit{ID},k}(\mathcal{D}) \\
\quad \tilde{\imath} \xleftarrow{\$} \{1,\ldots,T\} & \quad \tilde{\imath} \xleftarrow{\$} \{1,\ldots,T\} \\
\quad (pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)\,;\, sk_{\tilde{\imath}+1} \xleftarrow{\$} \mathsf{Update}^{\tilde{\imath}}(sk_1) & \quad (pk, sk_{\tilde{\imath}+1}) \xleftarrow{\$} \mathsf{LKG}(1^k, 1^T, \tilde{\imath}) \\
\quad \text{return } \mathcal{D}(pk, sk_{\tilde{\imath}+1}) & \quad \text{return } \mathcal{D}(pk, sk_{\tilde{\imath}+1})
\end{array}$$

   $\mathcal{D}$ is said to (t, ε)-solve the key indistinguishability problem if it runs in time t and

$$\left|\Pr\left[\mathbf{Exp}^{\text{ind-keys-real}}_{\mathit{ID},k}(\mathcal{D}) = 1\right] - \Pr\left[\mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathit{ID},k}(\mathcal{D}) = 1\right]\right| \geq \varepsilon.$$

   Furthermore, we say that ID is (t, ε)-key-indistinguishable, if no algorithm (t, ε)-solves the key-indistinguishability problem.

4. **Lossiness.** Let $\mathcal{I}_i$ be an impersonator for period i ($i \in \{1,\ldots,T\}$), st be its state. We consider the experiment $\mathbf{Exp}^{\text{los-imp-pa}}_{\mathit{ID},k,i}(\mathcal{I}_i)$ played between $\mathcal{I}_i$ and a hypothetical challenger:

$$\begin{array}{l}
\mathbf{Exp}^{\text{los-imp-pa}}_{\mathit{ID},k,i}(\mathcal{I}_i) \\
\quad (pk, sk_{i+1}) \xleftarrow{\$} \mathsf{LKG}(1^k, 1^T, i)\,;\, (cmt, st) \xleftarrow{\$} \mathcal{I}_i(pk, sk_{i+1})\ \ ;\, ch \xleftarrow{\$} \mathcal{C}\,;\, rsp \xleftarrow{\$} \mathcal{I}_i(ch, st) \\
\quad \text{return } \mathsf{Ver}(pk, cmt, ch, rsp, i)
\end{array}$$

   The impersonator $\mathcal{I}_i$ is said to ε-solve the impersonation problem with respect to i-lossy public keys if $\Pr\left[\mathbf{Exp}^{\text{los-imp-pa}}_{\mathit{ID},k,i}(\mathcal{I}_i) = 1\right] \geq \varepsilon$. Furthermore, ID is said to be ε-lossy if, for any period $i \in \{1,\ldots,T\}$, no (computationally unrestricted) algorithm ε-solves the impersonation problem with respect to i-lossy keys.

In addition, the **commitment space** of ID has **min-entropy** at least β, if for every period i, every security parameter k, and all honestly generated keys $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k)$, every bit string cmt^*, if $sk_i = \mathsf{Update}^{i-1}(sk_1)$, then:

$$\left|\Pr\left[\, cmt = cmt^* \;\middle|\; (cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)\,\right]\right| \leq 2^{-\beta}.$$

Usually, an identification scheme is also required to be sound, i.e., for a period i chosen uniformly at random, it should not be possible for an adversary to run the protocol with a honest verifier and to make the latter accept, if the public key has been generated honestly ($(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$) and if the adversary is only given a secret key sk_{i+1}. But, in our case, this soundness property follows directly from the key indistinguishability and the lossiness properties.

We could also consider more general lossy identification schemes where the simulatability of transcripts is computational instead of being statistical. However, we opted for not doing so because our security reduction is not tight with respect to the simulatability of transcripts.

$\underline{\mathsf{KG}(1^k, 1^T)}$	$\underline{\mathsf{Update}(sk_i)}$	$\underline{\mathsf{Sign}(sk_i, M)}$	$\underline{\mathsf{Ver}(pk, (\sigma, i), M)}$
$(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$	$sk \leftarrow \mathsf{Update}(sk_i)$	$(cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)$	$(cmt, rsp) \leftarrow \sigma$
return (pk, sk_1)	return sk	$ch \leftarrow \mathsf{H}((cmt, M, i))$	$ch \leftarrow \mathsf{H}((cmt, M, i))$
		$rsp \xleftarrow{\$} \mathsf{Prove}(sk_i, cmt, ch, st)$	$d \leftarrow \mathsf{Ver}(pk, cmt, ch, rsp, i)$
		$\sigma \leftarrow (cmt, rsp)$	return d
		return (σ, i)	

Figure 3.1: Generalized Fiat-Shamir transform for forward-secure signature

We remark that, for $T = 1$, a key-evolving lossy identification scheme becomes a standard lossy identification scheme, described in [AFLT12].[7]

Finally, we say that $\mathcal{ID}$ is **response-unique** if the following holds either for every lossy public key or for every normal public key (or for both): for all periods $i \in \{1, \ldots, T\}$, for all bit strings cmt (which may or may not be a correctly generated commitment), and for all challenges ch, there exists at most one response rsp such that $\mathsf{Ver}(pk, cmt, ch, rsp, i) = 1$.

3.2 Generalized Fiat-Shamir Transform

The forward-secure signature schemes considered in this paper are built from a key-evolving identification scheme via a straightforward generalization of the Fiat-Shamir transform [FS87], depicted in Figure 3.1. More precisely, the signature for period i is just the signature obtained from a Fiat-Shamir transform with secret key $sk_i = \mathsf{Update}^{i-1}(sk_1)$ (with the period i included in the random oracle input).

Let $\mathcal{FS}[\mathcal{ID}] = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be the signature scheme obtained via this generalized Fiat-Shamir transform.

Main security theorem. The following theorem is a generalization of Theorem 1 in [AFLT12] to key-evolving schemes. If we set $T = 1$ in Theorem 3.1, we get the latter theorem with slightly improved bounds, since forward security for $T = 1$ reduces to the notion of strong unforgeability for signature schemes. For the sake of simplicity and contrary to [AFLT12] where the completeness property of the underlying lossy identification scheme was only assumed to hold statistically, we assume perfect completeness, as this is satisfied by all of the schemes that we consider.

Theorem 3.1 *Let $\mathcal{ID} = (\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \mathcal{C}, \mathsf{Ver})$ be a key-evolving lossy identification scheme whose commitment space has min-entropy at least β (for every period i), let H be a hash function modeled as a random oracle, and let $\mathcal{FS}[\mathcal{ID}] = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be the signature scheme obtained via the generalized Fiat-Shamir transform (Figure 3.1). If $\mathcal{ID}$ is ε_s-simulatable, complete, (t', ε')-key-indistinguishable, and ε_ℓ-lossy, then $\mathcal{FS}[\mathcal{ID}]$ is $(t, q_h, q_s, \varepsilon)$-existentially-forward-secure in the random oracle model for:*

$$\varepsilon = T\ (\varepsilon' + (q_h + 1)\varepsilon_\ell) + q_s\varepsilon_s + (q_h + q_s + 1)q_s/2^\beta \qquad \text{and} \qquad t \approx t' - (q_s\, t_{\mathsf{Sim-Sign}} + (T-1)\, t_{\mathsf{Update}})$$

where $t_{\mathsf{Sim-Sign}}$ denotes the time to simulate a transcript using $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}$ and t_{Update} denotes the time to update a secret key using Update. Furthermore, if $\mathcal{ID}$ is response-unique, $\mathcal{FS}[\mathcal{ID}]$ is also $(t, q_h, q_s, \varepsilon)$-forward-secure.

The proof of Theorem 3.1 is an adaptation of the proof in [AFLT12] to the forward-security setting. As in [AFLT12], the main idea of the proof is to switch the public key of the signature scheme with a lossy one, for which forgeries are information-theoretically impossible with high probability. In our case, however, we need to guess the period i^* of the signature output by the adversary, in order to choose the correct type

[7] Contrary to the definition of lossiness given in [AFLT12], the impersonator $\mathcal{I}_1$ does not have access to an oracle $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,1,k}$ in $\mathbf{Exp}^{\text{los-imp-pa}}_{\mathcal{ID},k,1}(\mathcal{I}_1)$. However, we remark that this has no impact on the security definition as the execution of $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,1,k}$ does not require any secret information.

of lossy key to be used in the reduction and this is why we lose a factor T in the reduction. Moreover, as in [AFLT12], signatures queries are easy to answer thanks to the simulatability of the identification scheme. Finally, similarly to [AFLT12] and contrary to [AABN02], we remark that the factor q_H only multiply terms which are statistically negligible and, hence, they have no effect on the tightness of the proof.

Proof: Let us suppose there exists an adversary $\mathcal{A}$ which $(t, q_h, q_s, \varepsilon)$-breaks the existential forward security of $\mathcal{FS}$. Let us consider the games $\mathrm{G}_0, \ldots, \mathrm{G}_9$ of Figures 3.2 and 3.3. The random oracle $\mathbf{H}$ is simulated using a table HT containing all the previous queries to the oracle and its responses.

Before describing precisely all the games and formally showing that two consecutive games are indistinguishable, let us give a high-level overview of these games. The first game G_0 corresponds to the original security notion. We then change the way signatures are computed: instead of getting the challenge ch from the random oracle after generating the commitment cmt, we first choose it and then program the random oracle. We deal with programming and the possible collisions that it could generate in the games G_1, G_2, and G_3. We then simulate all signatures using the transcript simulator $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,i,k}$ in the game G_4, as ch can now be chosen independently of cmt. From this point on, the secret keys are only used to answer the **Break-In** queries. Hence, we can now guess the time period i^* of the forgery, abort if this is not guessed correctly, and generate a lossy public key for period $i^* + 1$ onwards (in the games G_6, G_7, and G_8). This makes us lose a factor T in the reduction. Finally, the lossiness property ensures that the adversary cannot generate a valid signature with non-negligible probability.

Let us now provide the proof details. First, we will assume that the set of queries to the random oracle made by the adversary always contains the query (cmt^*, M^*). This is without loss of generality because, given any adversary, we can always create an adversary (with the same success probability and approximately the same running time) that performs this query before calling **Finalize**. It only increases the total amount of hash queries by 1.

G_0 corresponds to a slightly stronger game than the game $\mathbf{Exp}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ defining the existential forward security of a key-evolving signature built from a key-evolving scheme via generalized Fiat-Shamir transform. We only force the forgery to be such that (cmt^*, M^*, i^*) is different from all the previous queries to the signing oracle, instead of just (M^*, i^*) being different from all the previous queries. This corresponds to a security notion stronger than existential forward-security but still weaker than strong forward-security (where we have to consider (cmt^*, rsp^*, M^*, i^*)).

In G_0, we have inlined the code of the random oracle in the procedure Sign, and we set bad whenever $\mathbf{H}((cmt, M, i))$ is already defined. We have also modified the code of the random oracle $\mathbf{H}$ such that the fp-th query (the critical query eventually related to the forgery) is answered by ch^*, a random challenge chosen in **Initialize**, where fp is a random integer in the range $\{1, \ldots, q_h + 1\}$. These modifications do not change the output of the original game.

To compute the probability $\Pr[\mathrm{G}_0(\mathcal{A}) \text{ sets } \mathsf{bad}]$, we remark that, for each signing query, the probability that there is a collision (i.e., bad is set for this query) is at most $(q_h + q_s + 1)/2^\beta$. By the union bound, we have $\Pr[\mathrm{G}_0(\mathcal{A}) \text{ sets } \mathsf{bad}] \leq (q_h + q_s + 1)q_s/2^\beta$.

In G_1, when bad is set, a new random value for $\mathbf{H}((cmt, M, i))$ is set in **Sign**. Since G_0 and G_1 are identical until bad, thanks to Lemma 2.1, we have $\Pr[\mathrm{G}_0(\mathcal{A}) \Rightarrow 1] - \Pr[\mathrm{G}_1(\mathcal{A}) \Rightarrow 1] \leq \Pr[\mathrm{G}_0(\mathcal{A}) \text{ sets } \mathsf{bad}] \leq (q_s + 1)q_s/2^\beta$.

In G_2, bad is no more set and the procedure **Sign** is rewritten in an equivalent way. Since the latter does not change the output of the game, we have $\Pr[\mathrm{G}_1(\mathcal{A}) \Rightarrow 1] = \Pr[\mathrm{G}_2(\mathcal{A}) \Rightarrow 1]$.

In G_3, the procedure **Sign** is changed such that the values (cmt, ch, rsp) are computed using the transcript generation function $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk,k}$. Since the latter does not change the output of the game, we have

Initialize(k, T) Game $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3, \mathrm{G}_4$

001 $S, \mathsf{HT}, \mathsf{QT} \leftarrow []$
002 $\mathrm{hc} \leftarrow 1$; $\mathrm{b} \leftarrow T + 1$
003 $\mathrm{fp} \xleftarrow{\$} \{1, \ldots, q_h + 1\}$; $ch^* \xleftarrow{\$} \mathcal{C}$
004 $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$
005 for $i = 1, \ldots, T - 1$
006 $\quad sk_{i+1} \leftarrow \mathsf{Update}(sk_i)$
007 return (pk, T)

H(x) Game $\mathrm{G}_0, \ldots, \mathrm{G}_9$

011 if $\mathsf{HT}[x] = \perp$ then
012 $\quad \mathsf{QT}[\mathrm{hc}] \leftarrow x$
013 $\quad$ if $\mathrm{hc} \neq \mathrm{fp}$ then
014 $\quad\quad \mathsf{HT}[x] \xleftarrow{\$} \mathcal{C}$
015 $\quad$ else
016 $\quad\quad \mathsf{HT}[x] \xleftarrow{\$} ch^*$
017 $\mathrm{hc} \leftarrow \mathrm{hc} + 1$
018 return $\mathsf{HT}[x]$

Break-In(i) Game $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3, \mathrm{G}_4$

021 if $1 \leq i \leq T$ then
022 $\quad \mathrm{b} \leftarrow \min(i, \mathrm{b})$
023 $\quad$ return sk_i
024 else
025 $\quad$ return $\perp$

Sign(M, i) Game G_0, $\boxed{\mathrm{G}_1}$

031 $(cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)$
032 if $\mathsf{HT}[(cmt, M, i)] \neq \perp$ then
033 $\quad \mathsf{bad} \leftarrow \mathsf{true}$
034 $\quad \boxed{\mathsf{HT}[(cmt, M, i)] \xleftarrow{\$} \mathcal{C}}$
035 else
036 $\quad \mathsf{HT}[(cmt, M, i)] \xleftarrow{\$} \mathcal{C}$
037 $ch \xleftarrow{\$} \mathsf{HT}[(cmt, M, i)]$
038 $rsp \xleftarrow{\$} \mathsf{Prove}(sk_i, cmt, ch, st)$
039 $\sigma \leftarrow (cmt, rsp)$
040 $S[(cmt, M, i)] \leftarrow rsp$
041 return (σ, i)

Sign(M, i) Game G_2

231 $(cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)$
232 $ch \xleftarrow{\$} \mathcal{C}$
233 $\mathsf{HT}[(cmt, M, i)] \leftarrow ch$
234 $rsp \xleftarrow{\$} \mathsf{Prove}(sk_i, cmt, ch, st)$
235 $\sigma \leftarrow (cmt, rsp)$
236 $S[(cmt, M, i)] \leftarrow rsp$
237 return (σ, i)

Sign(M, i) Game G_3

331 $(cmt, ch, rsp) \xleftarrow{\$} \mathsf{Tr}^{ID}_{pk, sk_i, k}$
332 $\mathsf{HT}[(cmt, M, i)] \leftarrow ch$
333 $\sigma \leftarrow (cmt, rsp)$
334 $S[(cmt, M, i)] \leftarrow rsp$
335 return (σ, i)

Sign(M, i) Game $\mathrm{G}_4, \ldots, \mathrm{G}_9$

431 $(cmt, ch, rsp) \xleftarrow{\$} \widetilde{\mathsf{Tr}}^{ID}_{pk, i, k}$
432 $\mathsf{HT}[(cmt, M, i)] \leftarrow ch$
433 $\sigma \leftarrow (cmt, rsp)$
434 $S[(cmt, M, i)] \leftarrow rsp$
435 return (σ, i)

Finalize$(M^*, (\sigma^*, i^*))$ Game $\mathrm{G}_0, \mathrm{G}_1, \mathrm{G}_2, \mathrm{G}_3, \mathrm{G}_4$

051 $d \leftarrow \mathsf{Ver}(pk, (\sigma^*, i^*), M^*)$
052 if $i^* \geq \mathrm{b}$ then
053 $\quad d \leftarrow 0$
054 $(cmt^*, rsp^*) \leftarrow \sigma^*$
055 if $S[(cmt^*, M^*, i^*)] \neq \perp$ then
056 $\quad d \leftarrow 0$
057 return d

Figure 3.2: Games $\mathrm{G}_0, \ldots, \mathrm{G}_4$ for proof of Theorem 3.1. G_1 includes the boxed code at line 034 but G_0 does not.

Initialize(k, T) Game G_5, G_6, G_7

501 $S, \mathsf{HT}, \mathsf{QT} \leftarrow []$
502 $\mathrm{hc} \leftarrow 1$; $\mathrm{b} \leftarrow T + 1$
503 $\mathrm{fp} \xleftarrow{\$} \{1, \ldots, q_h + 1\}$; $ch^* \xleftarrow{\$} \mathcal{C}$
504 $\tilde{\imath} \xleftarrow{\$} \{1, \ldots, T\}$
505 $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$
506 for $i = 1, \ldots, T - 1$
507 $\quad sk_{i+1} \leftarrow \mathsf{Update}(sk_i)$
508 return (pk, T)

Initialize(k, T) Game G_8, G_9

801 $\mathrm{hc} \leftarrow 1$; $\mathrm{b} \leftarrow T + 1$
802 $\mathrm{fp} \xleftarrow{\$} \{1, \ldots, q_h + 1\}$; $ch^* \mathcal{C}$
803 $\tilde{\imath} \xleftarrow{\$} \{1, \ldots, T\}$
804 $(pk, sk_{\tilde{\imath}+1}) \xleftarrow{\$} \mathsf{LKG}(1^k, 1^T, \tilde{\imath})$
805 for $i = 1, \ldots, \tilde{\imath}$
806 $\quad sk_i \leftarrow \perp$
807 for $i = \tilde{\imath} + 1, \ldots, T - 1$
808 $\quad sk_{i+1} \leftarrow \mathsf{Update}(sk_i)$
809 return (pk, T)

H(x) Game $G_0, \ldots, G_9$

011 if $\mathsf{HT}[x] = \perp$ then
012 $\quad \mathsf{QT}[\mathrm{hc}] \leftarrow x$
013 $\quad$ if $\mathrm{hc} \neq \mathrm{fp}$ then
014 $\quad\quad \mathsf{HT}[x] \xleftarrow{\$} \mathcal{C}$
015 $\quad$ else
016 $\quad\quad \mathsf{HT}[x] \xleftarrow{\$} ch^*$
017 $\mathrm{hc} \leftarrow \mathrm{hc} + 1$
018 return $\mathsf{HT}[x]$

Break-In(i) Game $G_5, \boxed{G_6}$

521 if $1 \leq i \leq T$ then
522 $\quad \mathrm{b} \leftarrow \min(i, \mathrm{b})$
523 $\quad$ if $i \leq \tilde{\imath}$
524 $\quad\quad \mathsf{bad} \leftarrow \mathsf{true}$
525 $\quad\quad \boxed{\text{return } \perp}$
526 $\quad$ return sk_i
527 else
528 $\quad$ return $\perp$

Break-In(i) Game G_7, G_8, G_9

721 if $1 \leq i \leq T$ then
722 $\quad \mathrm{b} \leftarrow \min(i, \mathrm{b})$
723 $\quad$ if $i \leq \tilde{\imath}$
724 $\quad\quad$ return $\perp$
725 $\quad$ return sk_i
726 else
727 $\quad$ return $\perp$

Sign(M, i) Game $G_4, \ldots, G_9$

431 $(cmt, ch, rsp) \xleftarrow{\$} \widetilde{\mathsf{Tr}}^{ID}_{pk,i,k}$
432 $\mathsf{HT}[(cmt, M, i)] \leftarrow ch$
433 $\sigma \leftarrow (cmt, rsp)$
434 $S[(cmt, M, i)] \leftarrow rsp$
435 return (σ, i)

Finalize$(M^*, (\sigma^*, i^*))$ Game $G_5, \boxed{G_6}$

551 $d \leftarrow \mathsf{Ver}(pk, (\sigma^*, i^*), M^*)$
552 if $i^* \geq \mathrm{b}$ then
553 $\quad d \leftarrow 0$
554 if $i^* \neq \tilde{\imath}$ then
555 $\quad \mathsf{bad} \leftarrow \mathsf{true}$
556 $\quad \boxed{d \leftarrow 0}$
557 $(cmt^*, rsp^*) \leftarrow \sigma^*$
558 if $S[(cmt^*, M^*, i^*)] \neq \perp$ then
559 $\quad d \leftarrow 0$
560 return d

Finalize$(M^*, (\sigma^*, i^*))$ Game $G_7, G_8, \boxed{G_9}$

751 $d \leftarrow \mathsf{Ver}(pk, (\sigma^*, i^*), M^*)$
752 if $i^* \geq \mathrm{b}$ or $i^* \neq \tilde{\imath}$ then
753 $\quad d \leftarrow 0$
754 $(cmt^*, rsp^*) \leftarrow \sigma^*$
755 if $\mathsf{QT}[\mathrm{fp}] \neq (cmt^*, M^*)$ then
756 $\quad \mathsf{bad} \leftarrow \mathsf{true}$
757 $\quad \boxed{d \leftarrow 0}$
758 if $S[(cmt^*, M^*, i^*)] \neq \perp$ then
759 $\quad d \leftarrow 0$
760 return d

Figure 3.3: Games $G_5, \ldots, G_9$ for proof of Theorem 3.1. G_6 includes the boxed code at lines 525 and 556 but G_5 does not; G_9 includes the boxed code at line 757 but G_7 and G_8 do not.

$\Pr[\,G_2(\mathcal{A}) \Rightarrow 1\,] = \Pr[\,G_3(\mathcal{A}) \Rightarrow 1\,]$.

In G_4, the q_s calls to the transcript generation function $\mathsf{Tr}^{I\!D}_{pk,sk_i,k}$ are replaced by q_s calls to the simulated transcript generation function $\widetilde{\mathsf{Tr}}^{I\!D}_{pk,i,k}$. Since the statistical distance between the distributions output by $\mathsf{Tr}^{I\!D}_{pk,sk_i,k}$ and by $\widetilde{\mathsf{Tr}}^{I\!D}_{pk,i,k}$ is at most ε_s, we have $\Pr[\,G_3(\mathcal{A}) \Rightarrow 1\,] - \Pr[\,G_4(\mathcal{A}) \Rightarrow 1\,] \leq q_s\varepsilon_s$.

In G_5, a period $\tilde{\imath} \in \{1, \ldots, T\}$ is chosen uniformly at random, and bad is set when the adversary queries **Break-In** with a period $\mathrm{b} \leq \tilde{\imath}$ or when the adversary outputs a signature for a period $i^* \neq \tilde{\imath}$. Since if G_5 outputs 1, we have $i^* < \mathrm{b}$, "bad is never set and G_5 outputs 1" (event $G_5(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_5$) if and only if "$i^* = \tilde{\imath}$ and G_5 outputs 1." Therefore, we have:

$$\Pr[\,G_5(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_5\,] = \sum_{i=1}^{T} \Pr[\,G_5(\mathcal{A}) \Rightarrow 1 \wedge i^* = i = \tilde{\imath}\,] = \frac{1}{T}\Pr[\,G_5(\mathcal{A}) \Rightarrow 1\,] = \frac{1}{T}\Pr[\,G_4(\mathcal{A}) \Rightarrow 1\,],$$

where the second equality comes from the fact that $\Pr[\,\tilde{\imath} = i\,] = \frac{1}{T}$ and that the event "$\tilde{\imath} = i$" is independent from the event "$G_5(\mathcal{A}) \Rightarrow 1 \wedge i^* = i$".

In G_6, the empty string $\perp$ is returned if **Break-In** is queried with a period $i \leq \tilde{\imath}$, and the game outputs 0 if $i^* \neq \tilde{\imath}$. Since G_5 and G_6 are identical until bad, according to Lemma 2.2, we have

$$\Pr[\,G_5(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_5\,] = \Pr[\,G_6(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_6\,] = \Pr[\,G_6(\mathcal{A}) \Rightarrow 1\,].$$

In G_7, some procedures have been rewritten in an equivalent way, and bad is now set when the query (cmt^*, M^*) is not the $\mathrm{fp}^{\mathrm{th}}$ query to the random oracle. Since the latter does not change the output of the experiment, we have $\Pr[\,G_6(\mathcal{A}) \Rightarrow 1\,] = \Pr[\,G_7(\mathcal{A}) \Rightarrow 1\,]$.

In G_8, the key is generated using the lossy key generation algorithm LKG for period $\tilde{\imath}$ instead of the normal key generation algorithm KG. From any adversary $\mathcal{A}$ able to distinguish G_7 from G_8, it is straightforward to construct an adversary which (t', ε'')-solves the key indistinguishability problem with $t' \approx t + (q_s t_{\mathsf{Sim-Sign}} + (T-1)t_{\mathsf{Update}})$ and $\varepsilon'' = |\Pr[\,G_7(\mathcal{A}) \Rightarrow 1\,] - \Pr[\,G_8(\mathcal{A}) \Rightarrow 1\,]|$. Therefore, thanks to the (t', ε')-key-indistinguishability of $I\!D$, if the adversary runs in time approximately at most $t' - (q_s t_{\mathsf{Sim-Sign}} + (T-1)t_{\mathsf{Update}})$:

$$\Pr[\,G_7(\mathcal{A}) \Rightarrow 1\,] - \Pr[\,G_8(\mathcal{A}) \Rightarrow 1\,] \leq \varepsilon'. \tag{3.1}$$

In G_9, the game outputs 0 if the signature does not correspond to the challenge ch^*. Since we have

$$\Pr[\,G_8(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_8\,] = \Pr[\,G_8(\mathcal{A}) \Rightarrow 1\,] \cdot \Pr[\,\mathsf{QT}[\mathrm{fp}] = (cmt^*, M^*)\,] = \frac{1}{q_h + 1}\Pr[\,G_8(\mathcal{A}) \Rightarrow 1\,],$$

and $\Pr[\,G_9(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_9\,] = \Pr[\,G_9(\mathcal{A}) \Rightarrow 1\,]$, according to Lemma 2.2, we have $\Pr[\,G_8(\mathcal{A}) \Rightarrow 1\,] = (q_h + 1)\Pr[\,G_9(\mathcal{A}) \Rightarrow 1\,]$.

From any adversary $\mathcal{A}$ for G_9, it is straightforward to construct an adversary $\mathcal{I}$ (not necessarily computationally bounded) which ε''-solves the impersonation problem with $\varepsilon'' = \Pr[\,G_9(\mathcal{A}) \Rightarrow 1\,]$. Therefore, we have $\Pr[\,G_9(\mathcal{A}) \Rightarrow 1\,] \leq \varepsilon_\ell$.

From the previous equalities and inequalities, we deduce that, for any adversary $\mathcal{A}$ running in time approximately at most $t' - (q_s t_{\mathsf{Sim-Sign}} + (T-1)t_{\mathsf{Update}})$:

$$\varepsilon \leq \Pr[\,G_0(\mathcal{A})\,] \leq T\ (\varepsilon' + (q_h + 1)\varepsilon_\ell) + q_s\varepsilon_s + (q_h + q_s + 1)q_s/2^\beta.$$

Let us now prove that $\mathcal{FS}$ is strongly forward-secure (with the same parameters) if $I\!D$ is response-unique. We first remark that, if we replace line 055 of G_0 in Figure 3.2 by

if $S[(cmt^*, M^*, i^*)] = rsp^*$ then

then we get exactly the game for forward security.

Therefore, if normal keys are response-unique, it is clear that this new game is equivalent to the original game G_0, since if $S[(cmt^*, M^*, i^*)]$ is defined, it is the only possible response rsp^*.

If lossy keys are response-unique, to prove forward security, it is sufficient to replace lines 055, 558, and 758 for games $\mathrm{G}_0, \ldots, \mathrm{G}_9$ in Figure 3.2 and Figure 3.3 by

if $S[(cmt^*, M^*, i^*)] = rsp^*$ then

Then the probability the adversary wins the new game G_9 is still bounded by ε_l since if $S[(cmt^*, M^*, i^*)]$ is defined, it is the only possible response rsp^*. ∎

Remark 3.2 As in the standard Fiat-Shamir transform, the signature obtained via the generalized transform consists of a commitment-response pair. However, in all schemes proposed in this paper, the commitment can be recovered from the challenge and the response. Hence, since the challenge is often shorter than the commitment, it is generally better to use the challenge-response pair as the signature in our schemes. Obviously, this change does not affect the security of our schemes.

Security theorems for comparisons with previous schemes. In the sequel, to be able to do a fair comparison, we also need the following variant of Theorem 3.1 and its associated straightforward corollary.

Theorem 3.3 *Let $\mathcal{ID} = (\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \mathcal{C}, \mathsf{Ver})$ be a key-evolving lossy identification scheme whose commitment space has min-entropy at least β (for every period i), let H be a hash function modeled as a random oracle, and let $\mathcal{FS}[\mathcal{ID}] = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be the signature scheme obtained via the generalized Fiat-Shamir transform. If $\mathcal{ID}$ is ε_s-simulatable, complete, (t', ε')-key-indistinguishable, and ε_ℓ-lossy, then $\mathcal{FS}[\mathcal{ID}]$ is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-existentially-forward-secure in the random oracle model for:*

$$t \approx \left(t' - (T-1)\, t_{\mathsf{Update}}\right) \cdot \left(\varepsilon - q_s\varepsilon_s - (q_h + q_s + 1)q_s/2^\beta\right) - q_s t_{\mathsf{Sim-Sign}}$$

as long as

$$\varepsilon > q_s\varepsilon_s + (q_h + q_s + 1)q_s/2^\beta \qquad \text{and} \qquad \varepsilon' \leq \delta\left(1 - \frac{1}{\mathrm{e}}\right) - \frac{(q_h+1)\,\varepsilon_\ell}{\varepsilon - q_s\varepsilon_s - (q_h + q_s + 1)q_s/2^\beta}$$

where where $t_{\mathsf{Sim-Sign}}$ denotes the time to simulate a transcript using $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}$, t_{Update} denotes the time to update a secret key using Update, and e (not to be confused with e) is the base of the natural logarithm. Furthermore, if $\mathcal{ID}$ is response-unique, $\mathcal{FS}[\mathcal{ID}]$ is also $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure.

Corollary 3.4 *Under the same hypothesis of Theorem 3.3, $\mathcal{FS}[\mathcal{ID}]$ is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-existentially-forward-secure in the random oracle model for:*

$$t \approx \frac{\left(t' - (T-1)\, t_{\mathsf{Update}}\right) \cdot \varepsilon}{2} - q_s t_{\mathsf{Sim-Sign}}$$

as long as

$$\varepsilon \geq 2\left(q_s\varepsilon_s + (q_h + q_s + 1)q_s/2^\beta\right) \qquad \text{and} \qquad \varepsilon' \leq \delta\left(1 - \frac{1}{\mathrm{e}}\right) - \frac{2\,(q_h+1)\,\varepsilon_\ell}{\varepsilon}.$$

Furthermore, if $\mathcal{ID}$ is response-unique, $\mathcal{FS}[\mathcal{ID}]$ is also $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure.

In concrete instantiations in the sequel, we often omit t_{Update} and $t_{\mathsf{Sim-Sign}}$ as these values are small compared to t', for any reasonable parameters. For any $\varepsilon > 0$ satisfying the above inequalities, under the assumption of Remark 2.4, we can say that the scheme $\mathcal{FS}[\mathcal{ID}]$ is about $(\frac{t'\varepsilon}{2}, q_h, q_s, T\varepsilon)$-forward-secure

(i.e., provide about $\log_2(t'/(2T))$ bits of security), if the underlying identification scheme $I\mathcal{D}$ is (about) $(t', (1-1/e)/2)$-hard. In other words, the security reduction loses a factor about T.

Proof of Corollary 3.4 from Theorem 3.3: It is a direct corollary of Theorem 3.3. The condition $\varepsilon > 2\left(q_s\varepsilon_s + (q_h+q_s+1)q_s/2^\beta\right)$ ensures that $\varepsilon - q_s\varepsilon_s - (q_h+q_s+1)q_s/2^\beta \geq \varepsilon/2$. ∎

Proof of Theorem 3.3: Let us suppose there exists an adversary $\mathcal{A}$ that $(t, q_h, q_s, \varepsilon, \delta)$-breaks $\mathcal{FS}[I\mathcal{D}]$. In particular, $\mathcal{A}$ $(t, q_h, q_s, \varepsilon\delta)$-breaks $\mathcal{FS}[I\mathcal{D}]$.

The proof of Theorem 3.3 is very similar to the proof of Theorem 3.1. We use the same games, except for **Initialize** and **Finalize** of games $\mathrm{G}_1, \ldots, \mathrm{G}_5$ which are replaced by the ones of game G_6. Indeed, in the game of the selective security, a period $\tilde{\imath}$ is chosen in **Initialize** and the adversary has to forge a signature for this period. Then the proof is identical except that $\Pr[\,\mathrm{G}_4(\mathcal{A}) \Rightarrow 1\,] = \Pr[\,\mathrm{G}_5(\mathcal{A}) \Rightarrow 1\,] = \Pr[\,\mathrm{G}_6(\mathcal{A}) \Rightarrow 1\,]$ and except for the inequality of Equation (3.1) (page 15).

We remark that, if we write $\gamma = \left(q_s\varepsilon_s + (q_h+q_s+1)q_s/2^\beta\right)$:

$$\Pr[\,\mathrm{G}_7(\mathcal{A}) \Rightarrow 1\,] \geq \varepsilon - \gamma \text{ with probability at least } \delta \text{ over } (pk, sk_1, \tilde{\imath}) \tag{3.2}$$

$$\Pr[\,\mathrm{G}_8(\mathcal{A}) \Rightarrow 1\,] \leq (q_h+1)\,\varepsilon_\ell \tag{3.3}$$

Let us construct an adversary $\mathcal{B}$ which (t'', ε'')-breaks the key indistinguishability property with $t'' \approx \frac{t+q_s t_{\mathsf{Sim-Sign}}}{\varepsilon-\gamma} + (T-1)t_{\mathsf{Update}}$ and $\varepsilon'' \geq \delta\left(1-\frac{1}{\mathrm{e}}\right) - \frac{1}{\varepsilon-\gamma}(q_h+1)\,\varepsilon_\ell$. $\mathcal{B}$ takes as input a period $\tilde{\imath}$, a public key pk and a secret key $sk_{\tilde{\imath}+1}$ for period $\tilde{\imath}+1$. It then runs $\mathcal{A}$ $\frac{1}{\varepsilon-\gamma}$ times and simulates the oracles as in game G_7 (or G_8, which is equivalent), except for **Initialize** where it uses directly its inputs $\tilde{\imath}$, pk and $sk_{\tilde{\imath}+1}$ (instead of picking them at random). If $\mathcal{A}$ outputs a correct forgery during one of its runs, $\mathcal{B}$ outputs 1. Otherwise, it outputs 0.

Clearly $\mathcal{B}$ perfectly simulates the environment of $\mathcal{A}$ in the game G_7, if pk is not lossy, or in the game G_8, if pk is lossy. According to Equation (3.2), if pk is not lossy, we have

$$\Pr[\,\mathcal{B} \Rightarrow 1 \mid pk \text{ normal}\,] \geq \delta \Pr\left[\mathcal{B} \Rightarrow 1 \;\middle|\; \begin{array}{l} pk \text{ normal and} \\ \Pr[\,\mathrm{G}_7(\mathcal{A}) \Rightarrow 1\,] \\ \geq \varepsilon - \gamma \end{array}\right] \geq \delta\left(1 - (1-(\varepsilon-\gamma))^{\frac{1}{\varepsilon-\gamma}}\right) \geq \delta\left(1-\frac{1}{\mathrm{e}}\right)$$

and, according to Equation (3.3), if pk is lossy, we have

$$\Pr[\,\mathcal{B} \Rightarrow 1 \mid pk \text{ lossy}\,] \leq \frac{1}{\varepsilon-\gamma}\Pr[\,\mathrm{G}_8(\mathcal{A}) \Rightarrow 1\,] \leq \frac{1}{\varepsilon-\gamma}(q_h+1)\,\varepsilon_\ell.$$

Therefore, the advantage of $\mathcal{B}$ is $\varepsilon'' \geq \delta\left(1-\frac{1}{\mathrm{e}}\right) - \frac{1}{\varepsilon-\gamma}(q_h+1)\,\varepsilon_\ell$. Its running time is $t'' \approx \frac{t+q_s t_{\mathsf{Sim-Sign}}}{\varepsilon-\gamma} + (T-1)t_{\mathsf{Update}}$. This proves the theorem. ∎

4 Tighter Security Reductions for Guillou-Quisquater-like Schemes

In this section, we prove tighter security reductions for the Guillou-Quisquater scheme (GQ, [GQ88]) and for a slight variant of the Itkis-Reyzin scheme (IR, [IR01]), which can also be seen as a forward-secure extension of the GQ scheme. We analyze the practical performance of this new scheme in the next section of this article.

4.1 Guillou-Quisquater Scheme

Let us describe the identification scheme corresponding to the GQ signature scheme,[8] before presenting our tight reduction and comparing it with the swap method.

SCHEME. Let N be a product of two distinct ℓ_N-bit primes p_1, p_2 and let e be a ℓ_e-bit prime, co-prime to $\phi(N) = (p_1 - 1)(p_2 - 1)$, chosen uniformly at random.[9] Let S be an element chosen uniformly at random in $\mathbb{Z}_N^*$ and let $U = S^e \bmod N$. Let $\mathfrak{c} = e \geq 2^{\ell_e - 1}$ and $\mathcal{C} = \{0, \ldots, \mathfrak{c} - 1\}$. The (normal) public key is $pk = (N, e, U)$ and the secret key is $sk = (N, e, S)$.

The goal of the identification scheme is to implicitly prove U is an e-residue.[10] The identification scheme is depicted in Figures 4.1 and 4.2. It works as follows. We recall that the scheme only supports one time-period (i.e., $T = 1$). Therefore, the Update algorithm is not needed and $sk_1 = sk$.First, the prover chooses a random element $R \in \mathbb{Z}_N^*$, computes $Y \leftarrow R^e \bmod N$. It sends Y to the verifier, which in turn chooses $c \in \{0, \ldots, \mathfrak{c} - 1\}$ and returns it to the prover. Upon receiving c, the prover computes $Z \leftarrow R \cdot S^c \bmod N$ and sends this value to the verifier. Finally, the verifier checks whether $Z \in \mathbb{Z}_N^*$ and $Z^e = Y \cdot U^c$ and accepts only in this case.[11]

The algorithm LKG chooses e and $N = p_1 p_2$ such that e divides $p_1 - 1$, instead of being co-prime to $\phi(N)$, and chooses U uniformly at random among the non-e-residue modulo N. The lossy public key is then $pk = (N, e, U)$. Proposition B.13 and Proposition B.16 show that if U is chosen uniformly at random in $\mathbb{Z}_N^*$, it is not an e-residue with probability $1 - 1/e$ and that it is possible to efficiently check whether U is an e-residue or not if the factorization of N is known: U is a e-residue if and only if, for any $j \in \{1, 2\}$, e does not divide $p_j - 1$ or $U^{(p_j - 1)/e} = 1 \bmod p_j$.

In the original scheme, any prime number e of large enough length ℓ_e could be used —to get negligible soundness or lossiness probability, ℓ_e needs to be at least equal to k. However, for our proof to work, we need the ϕ-hiding assumption to hold. This implies in addition that $\ell_e < \ell_N/4$.

SECURITY. Existing proofs for the GQ scheme lose a factor q_h in the reduction. In this section, we prove the previously described identification scheme ID is a lossy identification scheme, under the ϕ-hiding assumption. This yields a security proof of the strong unforgeability of the GQ scheme, with a tight reduction to this assumption.

More formally, we prove the following theorem:

Theorem 4.1 *The identification scheme ID depicted in Figures 4.1 and 4.2 is complete, perfectly simulatable, key-indistinguishable, $(1/\mathfrak{c})$-lossy, and response-unique. More precisely, if the ϕ-hiding problem is (t', ε')-hard, then the identification scheme ID is (t, ε)-key-indistinguishable for:*

$$t \approx t' \qquad \text{and} \qquad \varepsilon \leq \varepsilon' + \frac{1}{2^{\ell_e - 1}}.$$

Furthermore, the min-entropy β of the commitment space is $\log_2(\phi(N)) \geq \ell_N - 1$.

Thanks to Corollary 3.4 (with $T = 1$), we have the following corollary:

[8] There is a small difference between the GQ scheme described below and the original one in [GQ88]: in this paper, the distribution of e is assumed to be uniform over ℓ_e-bit primes, such that e does not divide $\phi(N)$, while in [GQ88] there was no restriction on e. See also Footnote 11.

[9] We choose e to be co-prime to $\phi(N)$ in this paper, as this makes the identification scheme response-unique and slightly simplifies the key indistinguishability proof. We point out the distribution of uniform ℓ_e-bit primes is $\frac{\ell_N + 1}{2^{\ell_e - 1}}$-close to the distribution of ℓ_e-bit primes that are co-prime to $\phi(N)$, according to Proposition B.19. Therefore, up to losing a negligible $\frac{\ell_N + 1}{2^{\ell_e - 1}}$ term in the security reduction, we can also take e to be a ℓ_e-bit prime.

[10] When e is co-prime with $\phi(N)$, any element $U \in \mathbb{Z}_N^*$ is an e-residue. However, under the ϕ-hiding assumption, this case is indistinguishable from the case where e divides $\phi(N)$, in which case only 1 out of e elements of $\mathbb{Z}_N^*$ is an e-residue. This is why we simply say that the goal of the identification scheme is to implicitly prove that U is an e-residue.

[11] The test $Z \in \mathbb{Z}_N^*$ can be replaced by the less expensive test $Z \bmod N \neq 0$, as explained in Section 6.2.

$\underline{\mathsf{KG}(1^k, 1)}$	$\underline{\mathsf{LKG}(1^k, 1, 1)}$
$e \xleftarrow{\$} \mathbb{P}_{\ell_e}$ $(N, p_1, p_2) \xleftarrow{\$} \mathsf{RSA}_{\ell_N}[\gcd(e, \phi(N)) = 1]$ $S \xleftarrow{\$} \mathbb{Z}_N^*$ $U \leftarrow S^e \bmod N$ $pk \leftarrow (N, e, U)$ $sk \leftarrow (N, e, S)$ return (pk, sk)	$e \xleftarrow{\$} \mathbb{P}_{\ell_e}$ $(N, p_1, p_2) \xleftarrow{\$} \mathsf{RSA}_{\ell_N}[p_1 = 1 \bmod e]$ $U \leftarrow \mathbb{Z}_N^* \setminus \mathsf{HR}_N[e]$ $pk \leftarrow (N, e, U)$ return $(pk, \perp)$

$\underline{\mathsf{Prove}(sk)}$
$(N, e, S) \leftarrow sk$
$st = R \xleftarrow{\$} \mathbb{Z}_N^*$
$cmt = Y \leftarrow R^e \bmod N$
return (cmt, st)
 	
$\underline{\mathsf{Prove}(sk, cmt, ch, st)}$
$(N, e, S) \leftarrow sk$
$R \leftarrow st$
$rsp = Z \leftarrow R \cdot S^c \bmod N$
return rsp
	$\underline{\mathsf{Ver}(pk, cmt, ch, rsp, 1)}$
	$(N, e, U) \leftarrow pk$
	$Y \leftarrow cmt$
	$c \leftarrow ch$
	$Z \leftarrow rsp$
	if $Z \in \mathbb{Z}_N^*$ and $Z^e = Y \cdot U^c$ then
	  return 1
	else
	  return 0

Figure 4.1: Formal description of the Guillou-Quisquater identification scheme (ℓ_e and ℓ_N are two parameters depending on k and $\mathfrak{c} = e \geq 2^{\ell_e - 1}$)

Theorem 4.2 *If the ϕ-hiding problem is (t', ε')-hard, then the GQ scheme is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure for $T = 1$ period in the random oracle model for:*

$$t \approx \frac{t' \cdot \varepsilon}{2}$$

as long as

$$\varepsilon \geq 2\frac{(q_h + q_s + 1)q_s}{2^{\ell_N - 1}} \qquad \text{and} \qquad \varepsilon' + \frac{1}{2^{\ell_e - 1}} \leq \delta\left(1 - \frac{1}{\mathrm{e}}\right) - \frac{2\,(q_h + 1)}{2^{\ell_e - 1}\,\varepsilon},$$

where e *(not to be confused with e) is the base of the natural logarithm.*

Under the assumption of Remark 2.4, we can say that the scheme is about $(\frac{t\varepsilon}{2}, q_h, q_s, \varepsilon)$-strongly-unforgeable if the ϕ-hiding problem is $(t, (1 - 1/e)/2)$-hard. This means roughly that if we want a k-bit security, the modulus has to correspond to a security level of $k' \approx k$ bits, which is tight.

Prover		Verifier
Input: $sk = (N, e, S)$		Input: $pk = (N, e, U)$
$R \xleftarrow{\$} \mathbb{Z}_N^*$ $Y \leftarrow R^e \bmod N$	$\xrightarrow{Y}$	
	$\xleftarrow{c}$	$c \xleftarrow{\$} \{0, \ldots, \mathfrak{c} - 1\}$
$Z \leftarrow R \cdot S^c \bmod N$	$\xrightarrow{Z}$	
		if $Z \notin \mathbb{Z}_N^*$ or $Z^e \neq Y \cdot U^c$ return reject return accept

Figure 4.2: Pictorial description of the Guillou-Quisquater identification scheme

Proof of Theorem 4.1: The proof that $I\mathcal{D}$ is **complete** follows immediately from the fact that, if $U = S^e \bmod N$, an honest execution of the protocol will always result in acceptance as $Z^e = (R \cdot S^c)^e = R^e \cdot (S^e)^c = Y \cdot U^c$.

Simulatability $I\mathcal{D}$ follows from the fact that, given $pk = (N, e, U)$, we can easily generate transcripts whose distribution is perfectly indistinguishable from the transcripts output by an honest execution of the protocol. This is done by choosing Z uniformly at random in $\mathbb{Z}_N^*$ and c uniformly at random in $\{0, \ldots, \mathfrak{c} - 1\}$, and setting $Y = Z^e/U^c$.

Let us prove the **key indistinguishability** property. When e is co-prime with $\phi(N)$, the function f defined by $f(x) = x^e \bmod N$ is a permutation over $\mathbb{Z}_N^*$ (see, e.g., Proposition B.14). Therefore, when $pk = (N, e, U) \xleftarrow{\$} \mathsf{KG}(1^k, 1)$, the element $U = S^e \in \mathbb{Z}_N^*$ (with $S \xleftarrow{\$} \mathbb{Z}_N^*$) is uniformly random. The distribution of normal public keys is therefore indistinguishable from the one where e divides $\phi(N)$ and U is chosen uniformly at random, according to the ϕ-hiding assumption. And in this latter distribution, U is not a e-residue with probability $1 - 1/e \geq 1 - 1/2^{\ell_e - 1}$ according to Proposition B.15, so this distribution is $(1/2^{\ell_e - 1})$-close to the distribution of lossy keys according to Lemma 2.3. Hence, we get the bound of the theorem.

To show that $I\mathcal{D}$ is **lossy**, we note that, when the public key is lossy, for every element Y chosen by the adversary, there exists only one value of $c \in \{0, \ldots, \mathfrak{c} - 1\}$ for which there exists a response Z that is considered valid by the verifier. To see why, assume for the sake of contradiction that there exist two different values c_1 and c_2 in $\{0, \ldots, \mathfrak{c} - 1\}$ for which there exists a valid response. Denote by Z_1 and Z_2 one of the valid responses in each case. Without loss of generality, assume that $c_1 < c_2$. Since $Z_1^e = Y \cdot U^{c_1}$ and $Z_2^e = Y \cdot U^{c_2}$, we have that $(Z_2/Z_1)^e = U^{c_2 - c_1}$. As $c_2 - c_1$ is a positive number smaller than $2^{\ell_e - 1}$, it is co-prime to e (since e is a prime and $e \geq 2^{\ell_e - 1}$). Therefore, according to Bézout's identity, there exists two integers u, v such that: $ue + v(c_2 - c_1) = 1$. So:

$$U = U^{ue + v(c_2 - c_1)} = (U^u)^e (U^{c_2 - c_1})^v = (U^u (Z_2/Z_1)^v)^e$$

and U is a e-residue, which is impossible. This means that the probability that a valid response Z_i exists in the case where U is not a e-residue is at most $1/\mathfrak{c}$. It follows that $I\mathcal{D}$ is $1/\mathfrak{c}$-lossy.

Response-uniqueness of $I\mathcal{D}$ follows from the fact that for any normal public key $pk = (N, e, U)$, e is co-prime with $\phi(N)$ and so the function f defined by $f(x) = x^e \bmod N$ is a permutation over $\mathbb{Z}_N^*$. For a commitment Y and a challenge c, the only response that is accepted by the verifier is therefore: $Z = (Y \cdot U^c)^{1/e}$. Note that since $Z \in \mathbb{Z}_N^*$ and $Z^e = Y \cdot U^c$, Y is also necessarily in $\mathbb{Z}_N^*$.

Finally, as for any normal public key $pk = (N, e, U)$, the function f defined by $f(x) = x^e \bmod N$ is a permutation over $\mathbb{Z}_N^*$, commitments R^e are uniformly random in $\mathbb{Z}_N^*$. Therefore the **commitment space** has **min-entropy** $\log_2(\phi(N)) \geq \ell_N - 1$. ▌

Comparison with the swap method. Applying the swap method [MR02] to the GQ identification scheme can also provide a signature with a tight reduction, to the RSA problem. But in this case, the signing algorithm needs to compute the e-root of the output of the random oracle modulo N. Therefore, instead of requiring two exponentiations modulo N with a ℓ_e-bit exponent, the signing algorithm requires one such exponentiation and one exponentiation modulo N with a ℓ_N-bit exponent. So, our signing algorithm is about $\ell_N/(2\ell_e)$ times faster, for the same parameters and the same security level, if we consider the ϕ-hiding problem is as hard as the RSA problem, and if we disregard the small differences of the exact tightness of the reductions. Furthermore, the swap method cannot be directly extended to the forward-secure extension of the GQ scheme, described in the next section, because the prover has to know the factorization of N.

4.2 Variant of the Itkis-Reyzin Scheme

SCHEME. The idea of this forward-secure extension of the GQ scheme consists in using a different e for each period. More precisely, let $e_1, \ldots, e_T$ be T distinct ℓ_e-bit primes chosen uniformly at random, among ℓ_e-bit primes that are co-prime to $\phi(N)$. Let $f_i = e_{i+1} \cdots e_T$, $f_T = 1$, and $E = e_1 \ldots e_T$. Let S be an element chosen uniformly at random in $\mathbb{Z}_N^*$ and let $U = S^E \bmod N$. Let $S_i = S^{E/e_i}$ and $S'_i = S^{E/f_i}$. Then the public key is $pk = (N, e_1, \ldots, e_T, U)$ and the secret key for period i is $sk_i = (N, e_i, \ldots, e_T, S_i, S'_i)$. We remark we can easily compute sk_{i+1} from sk_i, since $S_{i+1} = {S'_i}^{f_{i+1}} \bmod N$ and $S'_{i+1} = {S'_i}^{e_{i+1}} \bmod N$.

For period i, we have $S_i^{e_i} = U$ and the identification scheme works exactly as the previous one with public key $pk = (N, e_i, U)$ and secret key $sk = (N, e_i, S_i)$.

For the sake of simplicity, in this naive description of the scheme, we store the exponents $e_1, \ldots, e_T$ in both the public and secret keys. Therefore, the keys are linear in T, the number of periods. It is possible to have constant-size keys, either by using fixed exponents,[12] or by computing the exponents using a random oracle. This will be discussed in Section 5.1.

SECURITY. The security proof is similar to the one for the previous scheme, with the main difference being the description of the lossy key generation algorithm LKG. More precisely, on input $(1^k, 1^T, i)$, the algorithm LKG generates e_i and $N = p_1 p_2$ such that e_i divides $p_1 - 1$, instead of being co-prime to $\phi(N)$, and chooses U' uniformly at random among the non-e_i-residues modulo N. Then it chooses $T - 1$ distinct random ℓ_e-bit primes $e_1, \ldots, e_{i-1}, e_{i+1}, \ldots, e_T$, and sets $U = U'^{e_{i+1} \cdots e_T} \bmod N$, $S_{i+1} = U'^{e_{i+2} \cdots e_T} \bmod N$ and $S'_{i+1} = U'^{e_{i+1}} \bmod N$. The public key is $pk = (N, e_1, \ldots, e_T, U)$ and the secret key for period $i + 1$ is $sk_{i+1} = (N, e_{i+1}, \ldots, e_T, S_{i+1}, S'_{i+1})$ (or $\perp$ if $i = T$). We remark that, since U' is a non-e_i-residue, U is also a non-e_i-residue and so the public key pk is i-lossy.

More formally, using a similar analysis as for Theorem 4.1, we can prove the following theorem.

Theorem 4.3 *The identification scheme $\mathcal{ID}$ described above is complete, perfectly simulatable, key-indistinguishable, $(1/\mathfrak{c})$-lossy, and response-unique. More precisely, if the ϕ-hiding problem is (t', ε')-hard, then the identification scheme $\mathcal{ID}$ is (t, ε)-key-indistinguishable for:*

$$t \approx t' \qquad \text{and} \qquad \varepsilon \leq \varepsilon' + \frac{1}{2^{\ell_e - 1}}.$$

Furthermore, the min-entropy β of the commitment space is $\log_2(\phi(N)) \geq \ell_N - 1$.

Theorem 4.4 *If the ϕ-hiding problem is (t', ε')-hard, then our variant of the IR scheme is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure in the random oracle model for:*

$$t \approx \frac{t' \cdot \varepsilon}{2}$$

as long as

$$\varepsilon \geq 2\frac{(q_h + q_s + 1)q_s}{2^{\ell_N - 1}} \qquad \text{and} \qquad \varepsilon' + \frac{1}{2^{\ell_e - 1}} \leq \delta\left(1 - \frac{1}{\mathrm{e}}\right) - \frac{2\,(q_h + 1)}{2^{\ell_e - 1}\,\varepsilon},$$

where e *(not to be confused with e) is the base of the natural logarithm.*

Under the assumption of Remark 2.4, we can say that the scheme is about $(\frac{t\varepsilon}{2}, q_h, q_s, T\varepsilon)$-forward-secure if the ϕ-hiding problem is $(t, (1 - 1/e)/2)$-hard. This means roughly that if we want a k-bit security, the modulus has to correspond to a security level of $k' \approx k + \log_2(T)$ bits (k' being an approximate solution of $2^{k'} = \frac{2^k \varepsilon}{2} \frac{1}{T\varepsilon}$).

[12] However, if we use fixed exponents, we need to rely on a variant of the ϕ-hiding assumption, which uses fixed exponents instead of random ones. That is why we do not consider this solution in this paper.

$\underline{\mathsf{AlgH}'_{\mathsf{H}''}(i)}$
001 $found \leftarrow \mathsf{false}$
002 $cpt \leftarrow 0$
003 while $found \neq \mathsf{true}$
004 $\quad x \leftarrow \mathsf{H}''(i\|cpt)$
005 $\quad x \leftarrow x$ with LSB and MSB set to 1
006 $\quad$ if $\mathsf{isPrime}(x)$ then
007 $\quad\quad$ break
008 $\quad cpt \leftarrow cpt + 1$
009 return x

$\underline{\mathsf{AlgProgH}''_{\mathsf{H}'}(i\|cpt)}$
010 $cpt' \leftarrow 0$
011 while $\mathsf{HT}''(i\|cpt') \neq \mathsf{H}'(i)$
012 $\quad$ if $\mathsf{HT}''(i\|cpt') = \perp$ then
013 $\quad\quad y \xleftarrow{\$} \{0,1\}^{\ell_e}$
014 $\quad\quad y' \leftarrow y$ with LSB and MSB set to 1
015 $\quad\quad$ if $\mathsf{isPrime}(y')$ then
016 $\quad\quad\quad \mathsf{HT}''(i\|cpt') \leftarrow \mathsf{H}'(i)$
017 $\quad\quad\quad$ break
018 $\quad\quad$ else
019 $\quad\quad\quad \mathsf{HT}''(i\|cpt') \leftarrow y$
020 $\quad cpt' \leftarrow cpt' + 1$
021 if $cpt > cpt'$ and $\mathsf{HT}''(i\|cpt) = \perp$ then
022 $\quad \mathsf{HT}''(i\|cpt) \xleftarrow{\$} \{0,1\}^{\ell_e}$
023 return $\mathsf{HT}'(i\|cpt)$

Figure 5.1: Construction of a hash function $\mathsf{AlgH}'_{\mathsf{H}''} = \mathsf{H}'$ which outputs prime numbers, from a classical hash function H'' which outputs ℓ_e-bit string; and algorithm $\mathsf{AlgProgH}''$ which can simulate H'' in such a way that $\mathsf{AlgH}'_{\mathsf{AlgProgH}''_{\mathsf{H}'}} = \mathsf{H}'$. isPrime is a primality test. HT'' is a table which is initially empty.

5 Analysis of our Variant of the Itkis-Reyzin Scheme

In this section, we analyze our variant of the IR scheme and compare it with the original IR scheme [IR01] and the MMM scheme [MMM02].

5.1 Computation of the Exponents $e_1, \ldots, e_T$

As explained before, storing the exponents $e_1, \ldots, e_T$ in the keys is not a good idea since the key size becomes linear in T. Since we need $e_1, \ldots, e_T$ to be random primes to be able to do the reduction of key indistinguishability property to the ϕ-hiding assumption, we can use a second hash function H' (also modeled as a random oracle in the proof) that outputs prime numbers of length ℓ_e, and set $e_i = \mathsf{H}'(i)$.

Hash function for prime numbers which can be modeled as a random oracle. We can construct a hash function H' that outputs prime numbers of length ℓ_e, from a classical hash function H'' that only outputs ℓ_e-bit strings, as $\mathsf{H}'(i) = \mathsf{AlgH}'_{\mathsf{H}''}(i)$ where the algorithm AlgH' is depicted in Figure 5.1. This construction is close to the construction of a pseudorandom function (PRF) mapping to prime numbers in [HW09].

Furthermore, we remark that, if H'' is modeled as a random oracle, then so can $\mathsf{H}' = \mathsf{AlgH}'_{\mathsf{H}''}$. More precisely, we can simulate any experiment with a random oracle H' that outputs prime numbers, by an experiment with random oracle H'', by setting $\mathsf{H}'' = \mathsf{AlgProgH}''_{\mathsf{H}'}$, where the algorithm $\mathsf{AlgProgH}''$ is depicted in Figure 5.1. When the primality test isPrime is probabilistic (instead of being deterministic), the simulation might not be perfect but is still statistically indistinguishable.

For efficiency purposes, it is necessary to use a probabilistic primality test for isPrime, such as Miller-Rabin. Let us now study the **error probability of** AlgH'. To do that, let C denote the random variable equal to the number of primality tests needed in AlgH' (i.e., the final value of $cpt + 1$), if the primality tests are deterministic. According to Proposition B.18, C is a geometric random variable of parameter at least $1/(\ell_e - 1)$. So its expectation $\mathbf{E}[C]$, the average number of calls to isPrime, is at most $\ell_e - 1$. Let us suppose the error probability of the test (i.e., the probability a composite number is considered prime) is $\varepsilon_p = 2^{-\rho}$. In this case, the error probability of AlgH' for input i is at most

$$\varepsilon' = \sum_{j=0}^{\infty} \Pr\left[\begin{array}{l} C = j \wedge \mathsf{isPrime} \text{ has done an error on} \\ \mathsf{H}'(i\|0),\ \mathsf{H}'(i\|1),\ \ldots \text{or } \mathsf{H}'(i\|j) \end{array}\right] \leq \sum_{j=0}^{\infty} \Pr[C = j]\, j\, \varepsilon_p = \mathbf{E}[C]\, \varepsilon_p \leq (\ell_e - 1)\varepsilon_p.$$

Let us now analyze the **security of the resulting scheme** when e_i is set to $\mathsf{AlgH}'(i)$. We suppose that sk_i contains the exponent e_i. The secret key length is increased only by a small amount and the signing algorithm becomes faster, since it does not need to recompute e_i. For the sake of simplicity, we suppose KG, LKG, and Update use a deterministic algorithm $\mathsf{isPrime}'$ instead of isPrime for the generation of the exponents $e_i = \mathsf{AlgH}'(i)$.[13] In the resulting scheme, the only difference in the security proof for when the exponents $e_1, \ldots, e_T$ are stored in the public key is the following: the call to Ver in the procedure **Finalize** might compute an exponent $e_{i^*} = \mathsf{AlgH}'(i^*)$ different from the one computed by LKG. The use of a deterministic primality test for KG, LKG, and Update ensures that everywhere else is the security reduction, the exponents e_i are computed correctly. We do not want to do such an assumption for the primality test for Ver, as we want to optimize the speed of Ver as much as possible.

We can adapt the security proof by replacing isPrime in AlgH' in the verification in the procedure **Finalize** of the forward security game by a deterministic algorithm $\mathsf{isPrime}'$. This just adds a term $(\ell_e - 1)\varepsilon_p$ to the final probability for the adversary to win the original game.

Let us now analyze the **performance of** AlgH'. If we forget the probability of errors of the primality test and do not take into account the time to call H'',[14] the average time of AlgH' is $(\mathbf{E}[C]-1)t_{\mathsf{isPrime-composite}} + t_{\mathsf{isPrime-prime}}$, where $t_{\mathsf{isPrime-composite}}$ is the average running time of isPrime if its input is a composite number, and $t_{\mathsf{isPrime-prime}}$ is the average running time of isPrime if its input is a prime.

For Miller-Rabin test, if the input is prime, the algorithm roughly does $\rho/2$ exponentiations modulo an ℓ_e-bit number with an ℓ_e-bit exponent. Otherwise, if the input is a composite number, it does fewer than $4/3$ such exponentiations on average.[15] Therefore, $t_{\mathsf{isPrime-prime}} \approx \rho \frac{3}{2} \frac{1}{2} {\ell_e}^3$ and $t_{\mathsf{isPrime-composite}} \approx \frac{3}{2} \frac{4}{3} {\ell_e}^3$, therefore the total time is about $(\frac{3}{4}\rho + 2\ell_e){\ell_e}^3$.[16] In comparison, the time of a signature or a verification (if the exponents $e_1, \ldots, e_T$ are stored in the public key) is the time of two exponentiations with a modulus of length ℓ_N and an exponent of length ℓ_e, that means about $2\ell_N k^2$. A practical comparison can be found in Table 5.2.

5.2 Optimizations

In this section, we analyze optimizations of the original IR scheme and see that they can be applied to our scheme too. We also propose a specific optimization for our scheme.

CHECKING $Z \neq 0 \bmod N$ INSTEAD OF $Z \in \mathbb{Z}_N^*$ IN Ver.. As explained in Footnote 11, we can change the test $Z \in \mathbb{Z}_N^*$ in Ver by the test $Z \neq 0 \bmod N$. We suppose that this optimization is done in our comparison in Table 5.2.

e_i POWER OF SMALL PRIMES. If we slightly change the e_i to be power of small primes ε_i: $e_i = \varepsilon_i^{\ell_e / \lfloor \log(\varepsilon_i) \rfloor}$, we can make the generation of e_i faster since generating a small ${\ell_e}'$-bit prime ε_i is about $(\ell_e/{\ell_e}')^3$ faster than generating a ℓ_e-bit prime e_i. However, we need to change the ϕ-hiding assumption in order to be able to do the security reduction.[17]

PEBBLING. We also remark that the pebbling mechanism described in [IR01] can directly be applied to our scheme.

[13] They can also use a probabilistic algorithm $\mathsf{isPrime}''$ with error probability ε_p'' so that $T \cdot (\ell_e - 1) \cdot \varepsilon_p'' \leq 2^{-k-1}$. In that case, in the reduction, we would first replace $\mathsf{isPrime}''$ by a deterministic algorithm $\mathsf{isPrime}'$ in KG and Update. This would just add a term $2^{-secpar}$ to the final probability for the adversary to win the original game.

[14] In practice, H'' will be implemented using a hash function which is hundred times faster than any primality test.

[15] Actually, a Miller-Rabin test should be a little faster than an exponentiation since we can stop the exponentiation before the end, in some cases.

[16] The factor $\frac{3}{2}$ comes from the fact that in average, an exponentiation by a ℓ_e-bit number requires $\frac{3}{2}\ell_e$ multiplications, using the square-and-multiply algorithm.

[17] More precisely, we need e in the ϕ-assumption to be chosen as a power of a small prime number. The distribution of $N = p_1 p_2$ such that e divides p_1 can be sampled in polynomial time, if we assume the extended Riemann hypothesis (Conjecture 8.4.4 of [BS96]), exactly as when e is a prime. But to our knowledge, this new variant of the ϕ-hiding has not been analyzed and might actually not hold.

Table 5.1: Choice of parameters

k	q_h	q_s	ℓ_e	ε_p	ℓ_N
80	2^{80}	2^{30}	123	2^{-80}	≥ 1248
128	2^{128}	2^{46}	171	2^{-128}	≥ 3248

STORING *cpt*. Another possible trade-off consists in storing the last *cpt* of AlgH' for each i, in the public and secret keys. Since $\mathbf{E}[C] \leq \ell_e - 1$, the expected size of *cpt* is $\log_2 \ell_e$ and storing them increases the size of the keys by $T \log_2 \ell_e$. For small values of T this can be useful, since this completely removes the necessity of isPrime in Sign, Ver, and Update.

5.3 Choice of Parameters

In order to be able to compare the original IR scheme with our scheme, we need to choose various parameters. In Table 5.1, we show our choice of parameters for two security levels: $k = 80$ bits and $k = 128$ bits. When choosing these parameters, we considered a value of $T = 2^{20}$, as it enables to update the key every hour for up to 120 years. In both cases, $\varepsilon_p = 2^{-\rho}$ denotes the maximum error probability of the probabilistic primality test used in the random oracle for prime numbers H', whereas q_h and q_s specify the maximum number of queries to the random oracle and to the signing oracle, respectively, in the forward-security game. Choices of q_h and q_s comes from [MR02]: $q_h = 2^k$ because hash queries can be computed by the adversary itself, while q_s is much lower as each signing query needs to be answered by an honest user.

Let us explain our choice for ℓ_e. As in [MR02], we suppose $T\varepsilon, \delta \geq 2^{-20} \approx 10^{-6}$.[18] And we chose $\ell_e \approx k + 43$ to satisfy inequalities in security reductions in Theorem 4.4 and Theorem 5.1.[19] In the sequel, all the parameters are fixed except the length ℓ_N of the modulus.

5.4 Comparison with Existing Schemes

COMPARISON WITH THE ITKIS-REYZIN SCHEME. In this section, we compare the original IR scheme without optimization with our scheme (in which e_i is stored in the secret key sk_i, as in the IR scheme). The original IR scheme is very close to our scheme. The only differences are that the IR scheme requires that the factors p_1 and p_2 of the modulus N are safe primes[20] and that IR signatures for period i contain the used exponent e_i. Therefore the IR verification algorithm does not need to recompute the exponent, and is faster. In order to prevent an adversary from using an exponent for the break-in period to sign messages for an older period, the exponent has to be in a different set for each period. The security of the scheme comes from the strong RSA assumption. Unfortunately, we cannot use such an optimization with our security reduction for our scheme, because we need to know which exponent the adversary will use to make the key lossy for this exponent. But, as explained in Section 5.2, the other optimizations of the original IR scheme can also be applied to our scheme.

We first remark that for the same parameters k, ℓ_e, ℓ_N, our key generation algorithm is slightly faster since it does not require safe primes; and our signing and key update algorithms are as fast as the IR ones. The key and signature lengths of the signatures are nearly the same as the IR ones (IR signatures are only ℓ_e-bits longer than our signatures). The real difference is the verification time since our verification algorithm needs to recompute the e_i, contrary to the IR scheme. Verification consists of two exponentiations (modulo N with an ℓ_e-bit exponent) for the original scheme and two exponentiations

[18] We use $T\varepsilon$ instead of ε because of the way selective security notions are defined, see Remark 2.4 to understand this choice.

[19] The constant 43 comes from $-\log_2 \varepsilon - \log_2 \delta + 3$ (for our scheme) and $-\log_2 \varepsilon + \log_2 T + 3$ (for the original IR scheme).

[20] A safe prime p is an odd prime such that $(p-1)/2$ is also prime. This assumption is needed for the security reduction of the IR scheme.

Table 5.2: Benchmark (using parameters of Table 5.1)

		exponentiation[a]		prime generation[b]		sig. / verification orig.[c]		verification new[d]	
k	ℓ_N	op.[e]	ms[f]	op.[e]	ms[f]	op.[e]	ms[f]	op.[e]	ms[f]
k	ℓ_N	$\frac{3}{2}\,\ell_e\,{\ell_N}^2$	n/a	$(\frac{3}{4}\,\rho + 2\,\ell_e)\,{\ell_e}^3$	n/a	$3\,\ell_e\,{\ell_N}^2$	n/a	$3\,\ell_e\,{\ell_N}^2 + (\frac{3}{4}\,\rho + 2\,\ell_e)\,{\ell_e}^3$	n/a
(theoretical times)									
80	1248	$0.29 \cdot 10^9$	0.15	$0.68 \cdot 10^9$	0.26	$0.58 \cdot 10^9$	0.30	$1.26 \cdot 10^9$	0.56
(parameters chosen without considering exact security but just hardness of factoring N — insecure, should not be used)									
80	1920	$0.68 \cdot 10^9$	0.34	$0.68 \cdot 10^9$	0.26	$1.36 \cdot 10^9$	**0.68**	$2.04 \cdot 10^9$	**0.94**
(parameters chosen for our variant of the IR scheme considering exact security using our new reduction)									
80	6848	$8.65 \cdot 10^9$	3.09	$0.68 \cdot 10^9$	0.26	$17.3 \cdot 10^9$	**6.18**		
(parameters chosen for the original IR scheme considering exact security using the original reduction in [IR01])									

k: security parameter; ℓ_N: size of the modulus N; ℓ_e: size of the exponents; $\varepsilon_p = 2^{-\rho}$: probability of error of the Miller-Rabin probabilistic primality test, here $\rho = k$, see Section 5.1;
[a] time for an exponentiation modulo N with a ℓ_e-bit exponent; just for information, as this is one of the most important operation of the various schemes;
[b] time to generate the prime numbers e;
[c] verification time of the original scheme (also equal to the signature time for both schemes), estimated using the time of the two exponentiations;
[d] verification time of our scheme, estimated using the time of the two exponentiations and of the prime generation;
[e] approximate theoretical complexity;
[f] time on an Intel Core i5 750 (2.67 GHz), using GMP version 5.0.4 (`http://gmplib.org`), in our own implementation.

and an evaluation of the random prime oracle (roughly equivalent to a random prime generation) for our scheme.

Let us now focus on the exact security of the two schemes. As explained by Kakvi and Kiltz in [KK12], the best known attacks against the ϕ-hiding problems are the factorization of N. Let us also consider it is true for the strong RSA problem (since it just strengthens our result if it is not the case). We recall that currently, no reductions are known between the strong RSA and the ϕ-hiding problems.

In Appendix C.3, we show the following theorem, for the Itkis-Reyzin scheme:

Theorem 5.1 *If the strong RSA problem is (t', ε')-hard, then the previous scheme is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure in the random oracle model for:*

$$t \approx \frac{t' \cdot \varepsilon}{4q_h + 6}$$

as long as

$$\varepsilon \geq 2\left(\frac{(q_h + q_s + 1)q_s}{2^{\ell_N - 2\ell_e - 2}} + \frac{q_h + 1}{2^{\ell_e - 1}}\right) \qquad \text{and} \qquad \varepsilon' \leq \delta\left(1 - \frac{1}{\mathrm{e}}\right)^2,$$

where e *(not to be confused with e) is the base of the natural logarithm.*

For any $\varepsilon > 0$ satisfying the above inequalities, under the assumption of Remark 2.4, we can say that the scheme is about $(\frac{t'\varepsilon}{4q_h}, q_h, q_s, T\varepsilon)$-forward-secure (i.e., provide about $\log_2(t'/(4q_hT))$ bits of security) if the strong-RSA problem is (about) $(t', (1-1/e)^2/2)$-hard. This means that, if we want k bits of security and if we suppose that strong-RSA is as hard as factorization, then the modulus has to correspond to a security level of about $k' \approx k + \log_2(Tq_h)$ (k being an approximate solution of $2^{k'} = \frac{2^k\varepsilon}{4q_h}\frac{1}{T\varepsilon}$), compared to only $k' \approx k + \log_2(T)$ bits for our variant (see end of Section 4.2).

Therefore, with our choice of parameters, if we want $k = 80$ bits of security, we need to choose a

modulus length ℓ_N such that the factorization is $k + \log_2(T) = 100$-bit hard[21] (for our scheme) and $k+\log_2(Tq_h) = 180$-bit hard (for the original scheme). This corresponds to about $\ell_N \approx 1920$ and $\ell_N \approx 6848$ respectively, according to Ecrypt II [ECR11]. In this case, according to Table 5.2, our verification algorithm is about 6 times faster (0.94ms vs 6.18ms) and our signing algorithm is about 9 times faster (0.68ms vs 6.18ms). And our scheme generates 3.5 times shorter signatures.

Comparison with the MMM scheme. The MMM scheme [MMM02] is one of the most efficient generic constructions of forward-secure signatures (from any signature scheme), to the best of our knowledge. Furthermore, it does not require to fix the number of periods T. However, in the security proof, we have to bound the number of periods T the adversary can use (as query for the oracles **Sign** and **Break-In**). Its forward security can be reduced to the strong unforgeability of the underlying signature scheme with a loss of a factor T.

If we want to compare the MMM scheme with our variant of the IR scheme, the fairest solution is to instantiate the MMM scheme with the GQ scheme. Then we can use our tight reduction of the GQ scheme to the ϕ-hiding problem, to prove that the resulting MMM scheme is forward-secure with a relatively tight (losing only a factor T) reduction to the ϕ-hiding problem. In this setting, the MMM scheme and our scheme have approximately the same proven security. And the comparison of the MMM scheme with our scheme is roughly the same as the comparison in [MMM02] between the IR scheme and the MMM scheme (which did not take into account the tightness of the reduction).

Very roughly, we can say that the MMM key generation and key update algorithms are faster (about T times faster). However, MMM secret keys are longer. And, even if MMM public keys are shorter (more than 30 times for $k = 80, \ell_N = 1248$), in most cases, it is not really useful since signatures with the MMM scheme are about four times longer than signatures with our scheme ($4\ell_N + (\log k + \log T)k$ compared to $\ell_N + k$), and also about twice as long as the sum of the length of a public key of our scheme and a signature. Therefore, since the public key is used for verification, the total memory needed to store input data needed for the verification of a signature with the MMM scheme is still twice the amount of the one needed with our scheme. Furthermore, our scheme outperforms the MMM scheme with respect to verification time (considering Table 5.2, since the MMM verification algorithm verifies two classical GQ signatures). This means that, if verification time, signing time, and signature size are critical (for example, if verification or signing has to be performed on a smart-card), it would be more advantageous to use our scheme instead of the MMM scheme. More generally, our scheme tends to fare better than the MMM scheme if key updates are not performed as often and if T can be bounded by a reasonable constant. This would be the case, for example, if keys are updated once a day and the expected lifetime of the scheme is 3 years ($T = 2^{10}$), and key update time is not an important parameter.

6 Generic Factoring-Based Forward-Secure Signature Scheme

In this section, we show that all our previous results on the GQ scheme and its forward-secure extension can be generalized and applied to several other schemes. To do so, we first introduce a new generic factoring-based key-evolving lossy identification scheme and then show that several factoring-based signature and forward-secure signature schemes can be seen as simple instantiations of this generic scheme.

6.1 Generic Factoring-Based Forward-Secure Signature Scheme

Scheme. Let ℓ be a parameter, let N be an integer without small divisors, and let $e_1, \ldots, e_T$ be T integers and E be the least common multiple of $e_1, \ldots, e_T$. Let $S_1, \ldots, S_\ell$ be elements in $\mathbb{Z}_N^*$ and let

[21] We say that the factorization is n-bit hard if factorizing a given random RSA modulus with constant probability of success (for example 1/2) takes time at least 2^n. As usual when we want to choose parameters, we suppose that the strong RSA problem and the ϕ-hiding problems are as hard as factorization, as the best known algorithms to solve these problems just consist in factorizing the modulus N (for the parameters we have chosen).

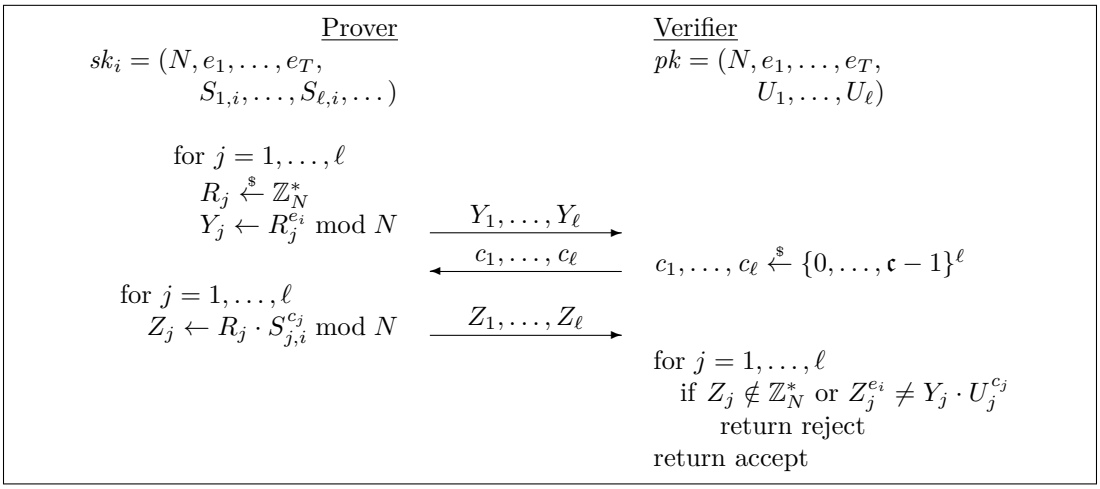

Figure 6.1: Description of the generic identification scheme $I\mathcal{D}$ for proving that the elements $U_1, \ldots, U_\ell$ in pk are all e_i-residues (for each $j \in \{1, \ldots, \ell\}$, $U_j = S_{j,i}^{e_i} \bmod N$).

$U_1, \ldots, U_\ell \in \mathbb{Z}_N^*$ be the corresponding E-powers. That is, for each $j \in \{1, \ldots, \ell\}$, $U_j = S_j^E \bmod N$. The public key is $pk = (N, e_1, \ldots, e_T, U_1, \ldots, U_\ell)$.[22] Let f_i be the least common multiple of $e_{i+1}, \ldots, e_T$ for each $i \in \{1, \ldots, T\}$ ($f_T = 1$) and let $S_{j,i} = S_j^{E/e_i}$ and $S'_{j,i} = S_j^{E/f_i}$, for each $1 \leq i \leq T$ and each $1 \leq j \leq \ell$. Then, the secret key for period $1 \leq i \leq T$ is $sk_i = (i, N, e_i, \ldots, e_T, S_{1,i}, \ldots, S_{\ell,i}, S'_{1,i}, \ldots, S'_{\ell,i})$. We remark that it is possible to compute sk_{i+1} from sk_i by computing: $S_{j,i+1} = S_{j,i}^{\prime f_i/e_{i+1}} \bmod N$ and $S'_{j,i+1} = S_{j,i}^{\prime f_i/f_{i+1}} \bmod N$.

The identification scheme is depicted in Figure 6.1 and is a straightforward extension of our variant of the IR scheme in Section 4.2. For period i, the prover's goal is to prove that the elements $U_1, \ldots, U_\ell$ are all e_i-residues. The scheme works as follows. First, the prover chooses an element $R_j \in \mathbb{Z}_N^*$ and computes $Y_j \leftarrow R_j^{e_i} \bmod N$, for $j \in \{1, \ldots, \ell\}$. It then sends $Y_1, \ldots, Y_\ell$ to the verifier, which in turn chooses $c_1, \ldots, c_\ell \in \{0, \ldots, \mathfrak{c} - 1\}^\ell$ (i.e., $\mathcal{C} = \{0, \ldots, \mathfrak{c} - 1\}^\ell$) and returns it to the prover. Upon receiving $c_1, \ldots, c_\ell$, the prover computes $Z_j \leftarrow R_j \cdot S_{j,i}^{c_j} \bmod N$ for $j \in \{1, \ldots, \ell\}$ and sends these values to the verifier. Finally, the verifier checks whether $Z_j \in \mathbb{Z}_N^*$ and $Z_j^{e_i} = Y_j \cdot U_j^{c_j}$ for $j \in \{1, \ldots, \ell\}$ and accepts only if this is the case.

As in the case of our variant of the IR scheme in Section 4.2, we store the exponents $e_1, \ldots, e_T$ in both the public and secret keys for the sake of simplicity. Nevertheless, in our concrete instantiations, it is possible to avoid storing these exponents either by computing them using a random oracle, as discussed in Section 5.1. Furthermore, in some cases, these exponents are a deterministic and easily computable function of the time period, and do not need to be stored in the public key.

Security. The proof of existential forward-security security uses the following condition:

Condition 6.1 *There exist a normal key generation algorithm* KG *and a lossy key generation algorithm* LKG *which take as input the security parameter k and the period i and output a pair (pk, sk'_{i+1}) such that, for every $i \in \{1, \ldots, T\}$:*

- *the pair (pk, sk'_{i+1}) generated by* LKG *is computationally indistinguishable from a pair (pk, sk_{i+1}) generated by* KG *and i calls to* Update *(to get sk_{i+1} from sk_1);*
- *when $(pk = (N, e_1, \ldots, e_T, U_1, \ldots, U_\ell),\ sk'_{i+1})$ is generated by* LKG*, for all $c \in \{0, \ldots, \mathfrak{c} - 1\}$, none*

[22] As for our variant of the IR scheme, we can use a hash function modeled as a random oracle (in the security proof) to avoid storing the exponents in the keys, as explained in Section 5.1.

of $U_1, \ldots, U_\ell$ is a $e'_{e_i,c,N}$-residue, where $e'_{e_i,c,N}$ is:

$$e'_{e_i,c,N} = \gcd_{j \in \{1,\ldots,m\}} \left(\frac{\gcd(e_i,\ (p_j^{k_j} - p_j^{k_j-1}))}{\gcd(c,\ e_i,\ (p_j^{k_j} - p_j^{k_j-1}))} \cdot e'_{i,j} \right),$$

with $N = p_1^{k_1} \ldots p_m^{k_m}$ being the prime decomposition of N and $e'_{i,j}$ being the greatest divisor of e_i co-prime to $p_i^{k_i} - p_i^{k_i-1}$.

We have the following security theorem:

Theorem 6.2 *Under Condition 6.1, $I\mathcal{D}$ is complete, perfectly simulatable, key-indistinguishable and $(1/\mathfrak{c}^\ell)$-lossy. Furthermore, the min-entropy β of the commitment scheme is at least the minimum over $i \in \{1, \ldots, T\}$ of $\ell \log_2(\phi(N, e_i))$ where $\phi(N, e_i)$ is the number of e_i-residues modulo N.*

Theorem 3.1 allows us to relate the existential forward-security of our generic key-evolving signature scheme $\mathcal{FS}$ to the security of the underlying identification scheme $I\mathcal{D}$. Thus our generic key-evolving signature scheme $\mathcal{FS}$ is existentially forward-secure under Condition 6.1.

Proof of Theorem 6.2: Let us do the proof for the case $T = 1$ and omit indices i to make the proof easier to understand. The key-evolving extension ($T > 1$) is straightforward.

Informally, the first part of the condition just corresponds key indistinguishability, whereas the second part of the condition corresponds to lossiness. More precisely, the second part ensures that, in a lossy setting, given a commitment, there cannot be more than one challenge for which there exists a response. This follows from some arithmetical results on residues, described in Appendix B.1, and namely in Theorem B.11. Formally, we have to show that $I\mathcal{D}$ meets the simulatability, completeness, key indistinguishability, and lossiness conditions.

The proof that $I\mathcal{D}$ is **complete** follows immediately from the fact that, if $U_j = S_j^e \bmod N$ for $j \in \{1, \ldots, \ell\}$, an honest execution of the protocol will always result in acceptance as $Z_j^e = (R_j \cdot S_j^{c_j})^e = R_i^e \cdot (S_j^e)^{c_j} = Y_j \cdot U_j^{c_j}$.

Simulatability of $I\mathcal{D}$ follows from the fact that, given $pk = (N, e, U_1, \ldots, U_\ell)$, we can easily generate transcripts whose distribution is perfectly indistinguishable from the transcripts output by an honest execution of the protocol. This is done by choosing Z_j uniformly at random in $\mathbb{Z}_N^*$ and c_j uniformly at random in $\{0, \ldots, \mathfrak{c} - 1\}$, and setting $Y_j = Z_j^e / U_j^{c_j}$ for $j \in \{1, \ldots, \ell\}$.

Key indistinguishability directly follows from Condition 6.1.

To show that $I\mathcal{D}$ is **lossy**, we note that, when the public key is lossy, for every element Y_j chosen by the adversary, there exists only one value of $c_j \in \{0, \ldots, \mathfrak{c} - 1\}$ for which there exists a valid response Z_j which passes the test. To see why, assume for the sake of contradiction that there exist two different values $c_{j,1}$ and $c_{j,2}$ in $\{0, \ldots, \mathfrak{c} - 1\}$ for which there exists a valid response. Denote by $Z_{j,1}$ and $Z_{j,2}$ one of the valid responses in each case. Without loss of generality, assume that $c_{j,1} < c_{j,2}$. Since $Z_{j,1}^e = Y_j \cdot U_i^{c_{j,1}}$ and $Z_{j,2}^e = Y_j \cdot U_j^{c_{j,2}}$, we have that $(Z_{j,2}/Z_{j,1})^e = U_j^{c_{j,2} - c_{j,1}}$. As $c_{j,2} - c_{j,1}$ is a positive number smaller than $\mathfrak{c}$, this means that U_j is an $e'_{e,c_{j,2}-c_{j,1},N}$-residue, according to Theorem B.11, which is a contradiction. This means that the probability that a valid response Z_j exists in the case where U_j is pseudo-e-residue is at most $1/\mathfrak{c}$. Since there are ℓ challenges, it follows that $I\mathcal{D}$ is ε_ℓ-lossy, with $\varepsilon_\ell = 1/\mathfrak{c}^\ell$. ∎

6.2 An Optimization

Let us present an optimization of the generic scheme for our cases. We consider the case of a classical signature scheme ($T = 1$) for the sake of simplicity.

We can remark that if the factorization[23] of N is hard, then we can replace the test $Z_j \in \mathbb{Z}_N^*$ by $Z_j \bmod N = 0$, in the identification scheme depicted in Figure 6.1. We just need to remark that the (existential) forward-security (or unforgeability) game with the original verification and the one with the new verification are identical until the following bad event happens: one of the Z_j is not equal to 0 modulo N, nor co-prime to N.

In all our schemes, knowing the factorization of N enables to solve key indistinguishability very efficiently (in polynomial time). Looking at the proof of Theorem 3.1, we remark that this optimization does not change the security bounds: we can indeed use the test $Z_j \bmod N = 0$ in Games G_0 to G_7, and then use the test $Z_j \in \mathbb{Z}_N^*$ instead for the following games. This does not change the bound in Equation (3.1) on page 15 for the reduction to key indistinguishability, as if $Z_j \bmod N = 0$ but $\mathbb{Z}_j \notin \mathbb{Z}_N^*$, we can factor N and directly solve key indistinguishability.

6.3 Instantiations

Guillou-Quisquater signature scheme. The case where $T = 1$, $\mathfrak{c} = e$ is an ℓ_e-bit prime number co-prime with $\phi(N)$, and $\ell = 1$ coincides with the GQ identification scheme recalled in Section 4.1. We already proved in Theorem 4.1, that this scheme $I\mathcal{D}$ is a lossy identification scheme. But let us now prove it again using Theorem 6.2, by showing that the scheme satisfies Condition 6.1.

We can prove the first part of Condition 6.1 (key indistinguishability of $I\mathcal{D}$), as in Theorem 4.1.

In addition, $e'_{e,c,N} = e$ for any $c \in \{1, \ldots, \mathfrak{c} - 1\}$. Indeed, if $\gcd(e, (p_j - 1)) = e$, $e'_j = 1$; otherwise $\gcd(e, (p_j - 1)) = 1$ and $e'_j = e$, because e is prime. Therefore $\gcd(e, (p_j - 1)) \cdot e'_j = e$ and $\gcd(e, c, (p_j - 1)) = 1$, for $j \in \{1, 2\}$. And U_1 is not a e-residue, when the public key is lossy. So, Condition 6.1 is satisfied.

Quadratic-residuosity-based signature scheme. The case where $e = \mathfrak{c} = 2$ and $T = 1$ is an important instantiation of the generic scheme as it coincides with the quadratic-residuosity-based scheme informally suggested by Katz and Wang in [KW03].

Suppose the algorithm LKG chooses $U_1, \ldots, U_\ell$ uniformly at random from the set $\mathsf{J}_N[e] \setminus \mathsf{HR}_N[e]$. Let us prove that Condition 6.1 is satisfied. To prove the key indistinguishability, we use the fact that the e-residuosity problem is random-self-reducible. That is, the distribution $(U_1, \ldots, U_\ell)$ where U_i is chosen uniformly at random from $\mathsf{HR}_N[e]$ is indistinguishable from the distribution $(U, U\alpha_2^e \bmod N, \ldots, U\alpha_\ell^e \bmod N)$ where U is chosen uniformly at random from $\mathsf{HR}_N[e]$ and α_i for $i \in \{2, \ldots, \ell\}$ is chosen uniformly at random from $\mathbb{Z}_N^*$. The latter distribution is clearly indistinguishable from the distribution $(U_1, \ldots, U_\ell)$ where U_i is chosen uniformly at random from $\mathsf{J}_N[e] \setminus \mathsf{HR}_N[e]$ due to the hardness of the e-residuosity problem. As a result, $I\mathcal{D}$ (t', ε')-key-indistinguishable where $t' \approx t$. Furthermore, $e'_{e,c,N} = 2$ for any $c \in \{1, \ldots, \mathfrak{c} - 1\}$ (i.e., $c = 1$), since $\gcd(e, (p_j - 1)) = 2$, $\gcd(e, c, (p_j - 1)) = 1$ and $e'_j = 1$, for $j \in \{1, 2\}$.

According to our security proof, this scheme is existentially unforgeable in the random oracle model based on the hardness of the quadratic-residuosity problem as long as ℓ is large enough to make the term $q_h/2^\ell$ negligible. And the reduction is tight.

2^t-root signature scheme by Ong and Schnorr. The case where $e = \mathfrak{c} = 2^t$, $\ell = 1$, and $T = 1$ coincides with the 2^t-root identification scheme by Ong and Schnorr [OS91]. Suppose $N = p_1 p_2$ is an RSA modulus such that 2^t divides $p_1 - 1$ and $p_2 - 1$, and the algorithm LKG chooses $U_1, \ldots, U_\ell$ uniformly at random from the set $\mathsf{J}_N[2] \setminus \mathsf{HR}_N[2]$. Let us prove that, if the gap 2^t-residuosity problem is hard, Condition 6.1 is satisfied.

Indeed, the key indistinguishability directly comes from the gap 2^t-residuosity. And, $e'_{e,c,N}$ is a multiple of 2 for any $c \in \{1, \ldots, \mathfrak{c} - 1\}$, since $\gcd(e, (p_j - 1)) = 2^t$, $\gcd(e, c, (p_j - 1))$ divides 2^{t-1} and $e'_j = 1$, for $j \in \{1, 2\}$. So, Condition 6.1 is satisfied.

According to our security proof, this scheme is existentially unforgeable in the random oracle model based on the hardness of the gap 2^t-residuosity problem as long as t is large enough to make the term

[23] By factorization, we mean finding any non-trivial factor of N.

$q_h/2^t$ negligible. And the reduction is tight.

We can easily extend this scheme to $\ell > 1$. The self-reducibility of the gap 2^t-residuosity problem enables to prove the key indistinguishability. In this case, we only need the term $q_h/2^{\ell t}$ to be negligible.

PAILLIER SIGNATURE SCHEME. The case where $\ell = 1$, $T = 1$, and $e = p_1 p_2$ is an RSA modulus, $N = e^2 = p_1^2 p_2^2$ and $\mathfrak{c} \leq \min(p_1, p_2)$ coincides with the Paillier signature scheme [Pai99].

Suppose $\mathfrak{c} \leq \min(p_1, p_2)$ (we can choose for example, $\mathfrak{c} = \lfloor \sqrt{e}/2 \rfloor$, if $p_1, p_2 \geq \sqrt{e}/2$) and the algorithm LKG chooses $U_1, \ldots, U_\ell$ uniformly at random from the set $\mathsf{J}_N[e] \setminus \mathsf{HR}_N[e]$. The proof of key indistinguishability is similar to the one of the above schemes.

In addition, $e'_{e,c,N} = p_1 p_2 = e$ for any $c \in \{1, \ldots, \mathfrak{c} - 1\}$. Indeed, if $\gcd(e, (p_j - 1)) = p_{3-j}$, $\gcd(e, (p_j^2 - p_j)) = p_1 p_2$ and $e'_j = 1$; otherwise $\gcd(e, (p_j^2 - p_j)) = p_j$ and $e'_j = p_{3-j}$. Therefore $\gcd(e, (p_j^2 - p_j)) \cdot e'_j = e$ and $\gcd(e,\ c,\ (p_j - 1)) = 1$, for $j \in \{1, 2\}$. So, Condition 6.1 is satisfied.

According to our security proof, the construction provides a signature scheme existentially unforgeable with a tight security reduction to the N-residuosity problem of [Pai99].

OUR VARIANT OF THE ITKIS-REYZIN SCHEME. The case where $T \geq 1$, e is a ℓ_e-bit prime number, and $\ell = 1$ coincides with our variant of the Itkis-Reyzin scheme in Section 4.2.

2^t-ROOT FORWARD-SECURE SIGNATURE SCHEME. The case in which $e_i = 2^{t(T-i+1)}$ with t a positive integer, $N = p_1 p_2$ is an RSA modulus such that 2^{tT} divides $p_1 - 1$ and $p_2 - 1$, and $\mathfrak{c} = 2^t$ is a generalization of the quadratic-residuosity-based scheme and the 2^t-root scheme. In this case, $f_i = e_i$, and we do not need to store $S'_{j,i}$ (for $j \in \{1, \ldots, \ell\}$.

The proof that Condition 6.1 is satisfied, is quite similar to the proof of the 2^t-root signature scheme by Ong and Schnorr above. To generate a lossy key for period $\tilde{\imath}$, LKG chooses $S_{1,\tilde{\imath}}, \ldots, S_{\ell,\tilde{\imath}}$ uniformly at random in $\mathsf{J}_N[2] \setminus \mathsf{HR}_N[2]$, $S_{j,i} = S_{j,\tilde{\imath}}^{e_{\tilde{\imath}}/e_i}$ for $i > \tilde{\imath}$, and $U_j = S_{j,\tilde{\imath}}^{e_{\tilde{\imath}}}$. We then remark that key indistinguishability can be trivially reduced to the key indistinguishability problem for the 2^t-root scheme by Ong and Schnorr in Section 6.3, which itself can be reduced to the gap $2^{t(T-\tilde{\imath}+1)}$-residuosity assumption. The lossiness can also be proven as for the 2^t-root scheme by Ong and Schnorr.

Therefore, this scheme is existentially forward-secure in the random oracle model based on the hardness of the gap 2^{ti}-residuosity assumption, for all $i \in \{1, \ldots, T\}$, as long as the exponent t and the parameter ℓ are large enough to make the term $q_h/2^{t\ell}$ negligible. And the reduction is relatively tight (we only lose a factor T).

Although this scheme appears to be new, it is of limited interest as its public key and secret key sizes are linear in the number T of time periods.

7 Impossibility Results on Tightness

Up to now, all the security proofs of forward security that we presented lose at least a factor T. In this section, we investigate whether such a loss in the reduction is inherent to the proposed schemes. Towards this goal, we show that any better reduction $\mathcal{R}$ from the forward security of a key-evolving signature scheme to a non-interactive hard problem Π can be converted into an efficient adversary against this hard problem for a large class of key-evolving schemes, which includes the previous schemes in this article. Therefore, the reduction for these schemes necessarily loses a factor T.

The idea of the proof is similar to the Coron's impossibility result in [Cor02, KK12]. After giving the intuition of the proof, we formally define the class of key-evolving schemes to which it applies, namely the key-verifiable schemes. Then, we specify the types of reductions in which we are interested, namely black-box non-rewinding reductions to non-interactive problems. Finally, we formally state the optimality result.

We recall that in an independent paper [BJLS16, Section 5.1], Bader et al. also studied signature schemes in a multi-user setting (with corruptions). Please refer to the introduction for more details.

7.1 Intuition

Our meta-reduction closely follows the one in [Cor02] and works roughly as follows. First, it chooses a random period $i^* \in \{1, \ldots, T\}$ and runs the reduction $\mathcal{R}$, asking for the secret key sk_{i^*} of period i^*. Then, it rewinds the reduction $\mathcal{R}$, asks for the secret key of period $i^* + 1$, and outputs a signature σ^* for the period i^* for a random message, using sk_{i^*}. Note that this signature is a valid forgery for the reduction, because, after rewinding, the break-in period is $i^* + 1 > i^*$. As a result, we have constructed an adversary for the problem Π, from the reduction $\mathcal{R}$, by simulating a forger.

However, this strategy does not work directly for two reasons. Firstly, this simulation is not perfect because it only outputs a forgery for periods i^* for which the reduction knows the secret key sk_{i^*}, whereas a real forger can output a forgery for periods for which the reduction knows or does not know the secret key sk_{i^*}. In fact, the idea behind the reductions for the schemes proposed in the previous sections is exactly to choose a random period i, create a bad key for this period i, and hope that the forger will output a signature for this period i. Actually this is the main reason why our meta-reduction does not prevent the existence of any reduction, but only the existence of reductions losing less than a factor T.

Secondly, there can be several different secret keys for a given public key and each of these secret keys may produce a different signature. Therefore, the signature σ^* generated by our adversary has not necessarily the same distribution as a signature which would have been generated by a real forger (who cannot rewind the reduction). That is why, the previous idea only works for a certain type of key-evolving signatures, which we call key-verifiable. Intuitively, a key-evolving signature scheme is said to be key-verifiable if one can check whether a given bit string is a secret key. Moreover, any secret key (or more precisely, any bit string) that passes the test produces the same signature distribution. This is the case in particular for all the schemes described in previous sections.

7.2 Key-Verifiable Key-Evolving Signature Scheme

More formally, a key-evolving signature scheme $\mathcal{FS}$ is ε_k-**key-verifiable**, if there exists a deterministic polynomial time algorithm VerK which takes as input a public key pk, a period i and a bit string and outputs a bit, such that, for any period i, for any public key pk:

- $\mathsf{VerK}(pk, i, sk_i) = 1$ for any real secret key sk_i,
- $\mathsf{VerK}(pk, i + j, \mathsf{Update}^j(\tilde{sk}_i)) = 1$ for any $j \in \{0, \ldots, T - i\}$, for any bit string $\tilde{sk}_i$ such that $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 1$,
- for any instantiation of the random oracle, for any message M, for any bit string $\tilde{sk}_i$, if we have $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 1$, then $\mathsf{Ver}(pk, \mathsf{Sign}(\tilde{sk}_i, M)) = 1$,
- for any $j \in \{0, \ldots, T - i\}$, for any instantiation of the random oracle, for all pk, $\tilde{sk}_i$, and $\tilde{sk'}_{i+j}$, if $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 1$ and $\mathsf{VerK}(pk, i + j, \tilde{sk'}_{i+j}) = 1$, then the two following distributions are ε_k-statistically indistinguishable:

$$\{(M, \sigma) \mid M \xleftarrow{\$} \mathcal{M}, \sigma \xleftarrow{\$} \mathsf{Sign}(\mathsf{Update}^j(\tilde{sk}_i), M)\}$$
$$\{(M, \sigma) \mid M \xleftarrow{\$} \mathcal{M}, \sigma \xleftarrow{\$} \mathsf{Sign}(\tilde{sk'}_{i+j}, M)\},$$

- $\mathsf{VerK}(pk, i, \perp) = 0$ for $i \in \{1, \ldots, T\}$, $\mathsf{VerK}(pk, T + 1, \perp) = 1$.

We remark that all the schemes described in previous sections are trivially 0-key-verifiable: to verify a secret key sk_i, we check whether $S_{j,i}^{e_i} = S_j$ and $S_{j,i}^{\prime f_i} = S'_j$ for all $j \in \{1, \ldots, \ell\}$.

7.3 Black-Box Non-Rewinding Reductions

We use the same formalization as Coron in [Cor02]. Let Π be a non-interactive (hard) problem, and $\mathcal{FS}$ be a key-evolving signature scheme. In our case, a non-interactive problem Π is just a set of instances and a function which maps an instance $\mathcal{I}$ (a bit string) to an answer $\mathcal{I}_a$. A problem Π is said (t, ε)-hard if no probabilistic adversary running in time t, which is given an instance $\mathcal{I}$ chosen uniformly at random, can output $\mathcal{I}_a$, with probability larger than ε.

A reduction algorithm $\mathcal{R}$ $(t_{\mathcal{R}}, q_h, q_s, \varepsilon_{\mathcal{A}}, \varepsilon_{\mathcal{R}})$-reduces the forward security of $\mathcal{FS}$ to the problem Π, if $\mathcal{R}$ takes as input a random instance $\mathcal{I}$ of Π, interacts with an adversary $\mathcal{A}$ which $(\cdot, q_h, q_s, \varepsilon_{\mathcal{A}})$-breaks the forward security of $\mathcal{FS}$, and outputs a solution to the instance $\mathcal{I}$ with probability at least $\varepsilon_{\mathcal{R}}$ and after at most $t_{\mathcal{R}}$ additional processing time.[24]

The interactions between the reduction and the adversary involve five types of queries: signature queries, break-in queries, random-oracle queries, initialization queries, and finalization queries. In other words, we only consider black-box reduction without rewinding: the reduction cannot access the code of the adversary nor can it rewind the adversary.

7.4 Main Theorem

In this section, we prove that, if a key-evolving signature scheme is key-verifiable, then any black-box non-rewinding reduction from its forward security to a hard problem Π can be used to solve Π with probability roughly greater than $\varepsilon_{\mathcal{R}} - \varepsilon_{\mathcal{A}}/T$. This means that if the success probability of the reduction $\varepsilon_{\mathcal{R}}$ is greater than $\varepsilon_{\mathcal{A}}/T$, then one can solve the hard problem. In particular, in our case, this shows that the security proof of our variant of the Itkis-Reyzin scheme is optimal. More formally, we prove the following theorem:

Theorem 7.1 *Let Π be a non-interactive hard problem, $\mathcal{FS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ be a ε_k-key-verifiable key-evolving signature scheme, and $\mathcal{R}$ be a black-box non-rewinding reduction algorithm (see Section 7.3). If $\mathcal{R}$ $(t_{\mathcal{R}}, q_h, q_s, \varepsilon_{\mathcal{A}}, \varepsilon_{\mathcal{R}})$-reduces the forward security of $\mathcal{FS}$ to the problem Π, then, from $\mathcal{R}$, we can build an algorithm $\mathcal{B}$ which (t, ε)-solves the hard problem Π, with*

$$t \approx 2t_{\mathcal{R}} + T\,(t_{\mathsf{Update}} + t_{\textbf{\textit{Break-In}}}) \qquad\qquad \varepsilon = \varepsilon_{\mathcal{R}} - \frac{\varepsilon_{\mathcal{A}}}{T} - \varepsilon_k.$$

*where t_{Update} is the time of Update of $\mathcal{FS}$ and $t_{\textbf{\textit{Break-In}}}$ is the time of the **Break-In** procedure of the reduction $\mathcal{R}$ (and so $t_{\textbf{\textit{Break-In}}} \leq t_{\mathcal{R}}$).*

We remark that this theorem also applies to existential forward security under key-only attack (i.e., $q_s = 0$) and also when the signature is in the standard model. This theorem also holds with respect to the original notion of forward security in [BM99].

Proof: Firstly, we remark that we can change $\mathcal{R}$ such that, in any given state of the reduction (in which **Break-In** has not been called), the outputs $\tilde{sk}_i$ of a **Break-In** request for periods i are so that, there exists $\tilde{\imath} \in \{1, \ldots, T\}$, such that for any $i > \tilde{\imath}$, $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 1$ and for any $i \leq \tilde{\imath}$, $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 0$. For that, it suffices to use the maximum $i \leq \tilde{\imath}$ such that $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 1$ as $\tilde{\imath}$, and then to output $\mathsf{Update}^{i-\tilde{\imath}}(\tilde{sk}_i)$ if $i \geq \tilde{\imath}$ and $\perp$ otherwise. This clearly does not decrease the success probability of the reduction, since any adversary can be transformed such that it outputs $\perp$ as soon as $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 0$. Furthermore, this only increases the time of $\mathcal{R}$ by at most $T\,(t_{\mathsf{Update}} + t_{\textbf{Break-In}})$. This transformation of $\mathcal{R}$ ensures that the probability that $\mathsf{VerK}(pk, i, \tilde{sk}_i) = 0$ and $\mathsf{VerK}(pk, i+1, \tilde{sk}_{i+1}) = 1$ is at most $1/T$, if i is chosen uniformly at random in $\{1, \ldots, T\}$.

We now consider the following (all powerful) adversary $\mathcal{A}$: it picks a period i^* uniformly at random in $\{1, \ldots, T\}$. Then it finds by brute-force a key $\tilde{sk}_{i^*}$ such that $\mathsf{VerK}(pk, i^*, \tilde{sk}_{i^*}) = 1$. He queries **Break-In**

[24] $t_{\mathcal{R}}$ does not include the time of the forger.

with period i^*+1. Let $\tilde{sk}_{i^*+1}$ be the output of **Break-In**. If $\mathsf{VerK}(pk, i^*+1, \tilde{sk}_{i^*+1}) = 0$, it stops and outputs $\perp$. Otherwise, it chooses a random message M^* (or a fixed message, it does not matter) and computes a signature σ^* on M^* using $\tilde{sk}_{i^*}$. Finally it outputs σ^* to $\mathcal{R}$ with probability $\varepsilon_{\mathcal{A}}$, and $\perp$ with probability $1-\varepsilon_{\mathcal{A}}$.

If the reduction $\mathcal{R}$ plays with such an adversary, it should solve the hard problem with probability $\varepsilon_{\mathcal{R}}$, by definition. Let us now describe an adversary $\mathcal{B}$ which has approximately the same external behavior (the reduction sees approximately the same distribution of queries) as $\mathcal{A}$; but which works in polynomial time by rewinding the reduction $\mathcal{R}$.

Firstly, $\mathcal{B}$ picks i^* uniformly at random in $\{1, \ldots, T\}$ and queries **Break-In** with period i^*. Let $\tilde{sk}_{i^*}$ be the output of **Break-In**. Then, $\mathcal{B}$ rewinds the reduction $\mathcal{R}$ before calling **Break-In**, and queries **Break-In** with period i^*+1. Let $\tilde{sk}_{i^*+1}$ be the output of **Break-In**. If $\mathsf{VerK}(pk, i^*+1, sk_{i^*+1}) = 0$, $\mathcal{B}$ submits the signature $\perp$ (since the reduction $\mathcal{R}$ has cheated). Otherwise, $\mathcal{B}$ chooses a random message M^*, compute a signature σ^* of M^* using $\tilde{sk}_{i^*}$ and submits the signature σ^* of M^* under period i^*, with probability $\varepsilon_{\mathcal{A}}$, and $\perp$ with probability $1-\varepsilon_{\mathcal{A}}$. We remark that the reduction sees that $\mathcal{B}$ does exactly the same queries as $\mathcal{A}$, except for the last query **Finalize** (the submission of the forged signature σ^*), because it does not see it has been rewound. Therefore, we just have to analyze the difference between the distribution of σ^*.

We remark that, thanks to the initial discussion, the probability ε_{Bad} that $\mathsf{VerK}(pk, i^*, \tilde{sk}_{i^*}) = 0$ and $\mathsf{VerK}(pk, i+1, \tilde{sk}_{i+1}) = 1$ is at most $\varepsilon_{Bad} \leq 1/T$. Let us call this case, the bad case. In the bad case, the behavior of $\mathcal{B}$ only differs from the one of $\mathcal{A}$, with probability $\varepsilon_{\mathcal{A}}$, because $\mathcal{A}$ outputs $\perp$, with probability $1-\varepsilon_{\mathcal{A}}$. Therefore, in the bad case, the probability the $\mathcal{R}$ solves Π is at least $\varepsilon_{\mathcal{R}} - \varepsilon_{\mathcal{A}}$.

Furthermore, in the good case, the signature produced is statistically indistinguishable from a signature produced using sk_i, since $\mathsf{VerK}(pk, i, sk_i) = 1$, and since $\mathcal{FS}$ is key-verifiable. Therefore, in this good case, the signature could have been produced by $\mathcal{A}$, and the reduction solves the hard problem with probability at least $\varepsilon_{\mathcal{R}} - \varepsilon_k$.

So, the total success probability of our adversary $\mathcal{B}$ for problem Π is at least $(\varepsilon_{\mathcal{R}} - \varepsilon_{\mathcal{A}}) \cdot \varepsilon_{Bad} + (\varepsilon_{\mathcal{R}} - \varepsilon_k) \cdot (1-\varepsilon_{Bad}) \geq \varepsilon_{\mathcal{R}} - \varepsilon_{\mathcal{A}} \cdot \varepsilon_{Bad} - \varepsilon_k \cdot (1-\varepsilon_{Bad}) \geq \varepsilon_{\mathcal{R}} - \varepsilon_{\mathcal{A}}/T - \varepsilon_k$. ∎

8 Multi-User and Tightly Forward-Secure Signature Schemes

In this section and in the next, we show how to circumvent the previous impossibility result, by constructing key-evolving signature schemes which are not key-verifiable and whose forward security can be tightly reduced to the underlying hard problem. In order to make the construction as clear as possible, we first introduce in Section 8.1 a new security notion for classical signature schemes, namely strong unforgeability in a multi-user setting with corruptions (M-SUF-CMA). Next, in Section 8.2, we present a transformation from M-SUF-CMA to forward-secure signature schemes which preserves tightness. Finally, in Section 9, we propose two constructions of M-SUF-CMA signature schemes with tight security reductions to the underlying hard problem.

8.1 M-SUF-CMA Signature Schemes

Informally the M-SUF-CMA security notion is close to the classical SUF-CMA security notion, except the adversary can dynamically ask for public keys pk, and for the associated secret key of any received public key pk (i.e., "corrupt" the public key pk), and wins if it forges a signature for a non-corrupted public key.

More formally, a signature scheme is $(t, I, q_h, q_s, \varepsilon)$-M-SUF-CMA, if, for any adversary A running in time at most t and making at most I queries to the key generation oracle **KG**, at most q_h queries to the random oracle, at most q_s queries to the signing oracle:

$$\mathbf{Adv}^{\text{m-suf-cma}}_{\mathcal{DS},k,I}(\mathcal{A}) = \Pr\left[\mathbf{Exp}^{\text{m-suf-cma}}_{\mathcal{DS},k,I}(\mathcal{A})\right] \leq \varepsilon,$$

Game $\mathbf{Exp}^{\text{m-suf-cma}}_{\mathcal{DS},k}(\mathcal{A})$

Initialize(1^k)	**KG**(par)	**Corrupt**(pk)
$S \leftarrow \emptyset$ $P \leftarrow []$ $par \xleftarrow{\$} \mathsf{PG}(1^k)$ for $i = 1, \ldots, I$ $(pk_i, sk_i) \leftarrow \mathsf{KG}(par)$ return par	$(pk, sk) \xleftarrow{\$} \mathsf{KG}(par)$ $P[pk] \leftarrow sk$ return pk	$sk \leftarrow P[pk]$ $P[pk] \leftarrow \perp$ return sk
Sign(M, pk)		**Finalize**(pk^*, M^*, σ^*)
$sk \leftarrow P[pk]$ if $sk = \perp$ then return $\perp$ $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, M)$ $S \leftarrow S \cup \{(pk, M, \sigma)\}$ return σ		$d \leftarrow \mathsf{Ver}(pk^*, \sigma^*, M^*)$ if $(pk^*, M^*, \sigma^*) \in S$ then $d \leftarrow 0$ if $P[pk^*] = \perp$ then $d \leftarrow 0$ return d

Figure 8.1: Game defining the M-SUF-CMA security of a signature scheme $\mathcal{DS} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$

where $\mathbf{Exp}^{\text{m-suf-cma}}_{\mathcal{DS},k,I}(\mathcal{A})$ is the game depicted in Figure 8.1. In this definition, we also include an additional algorithm PG that takes 1^k as input and generates common parameters par for the signature scheme. The common parameters par are also given as input to KG together with the security parameter 1^k.

This new definition is quite different from the definition of multi-user security introduced by Menezes and Smart in [MS04], since, on the one hand, we do not take into account the key substitution attack (and so in this way, our notion is weaker) but, on the other hand, we allow the corruption of any entity (and in this way, our notion is stronger).

Link with SUF-CMA. It is straightforward to see that any $(t, q_h, q_s, \varepsilon)$-SUF-CMA signature scheme is also a $(t, I, q_h, q_s, \varepsilon/I)$-M-SUF-CMA signature scheme. The reduction just consists in guessing the index of the public key of the forged signature.

We remark the M-SUF-CMA notion is somewhat more realistic than SUF-CMA, since a signature scheme is never used by only one user in practice. For example, if a large company uses I signature keys and that some of these keys may have been compromised by an adversary, this company may want to ensure that this adversary will be unable to produce valid signatures for any of the uncorrupted keys. But, if a scheme loses a factor I in the M-SUF-CMA security reduction, and if I is large enough, the security of the whole system can be affected (for example, if $I = 2^{40}$ and the expected level of security if 80 bits, then an attacker may be able to forge a signature in time 2^{40}, which is feasible). Therefore a tight M-SUF-CMA scheme is not only a tool to construct tight forward-secure scheme but can also be important in practice.

8.2 From M-SUF-CMA to Forward-Secure Signature Schemes

A naive generic construction of a forward-secure scheme $\mathcal{FS} = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ from a standard signature scheme $\mathcal{DS}' = (\mathsf{PG}', \mathsf{KG}', \mathsf{Sign}', \mathsf{Ver}')$, described in [And00, BM99] and depicted in Figure 8.2, is to simply use a different key pair (pk^i, sk^i) for each period i. Even though this construction has linear-sized public and secret keys, proving it secure in a tight manner is already not trivial. However, if $\mathcal{DS}'$ is $(t, I, q_s, q_h, \varepsilon)$-M-SUF-CMA, then it is straightforward to see that $\mathcal{FS}$ is $(t, q_s, q_h, \varepsilon)$-forward-secure for $T = I$ periods.

More efficient generic constructions of forward-secure signature schemes $\mathcal{FS}$ from standard SUF-CMA signature schemes $\mathcal{DS}'$ were proposed in [And00, BM99, Kra00, MMM02]. As in the case of the naive construction in Figure 8.2, the security of the forward-secure signature scheme in [And00] and of the binary certification tree construction in [BM99] can also be tightly reduced to the M-SUF-CMA security of $\mathcal{DS}'$,

$\underline{\mathsf{KG}(1^k, 1^T)}$	$\underline{\mathsf{Update}(sk_i)}$	$\underline{\mathsf{Sign}(sk_i, M)}$	$\underline{\mathsf{Ver}(pk, (\sigma, i), M)}$
$par \xleftarrow{\$} \mathsf{PG}(1^k)$	$(sk^i, \ldots, sk^T) \leftarrow sk_i$	$(sk^i, \ldots, sk^T) \leftarrow sk_i$	$(pk^1, \ldots, pk^T) \leftarrow pk$
for $i = 1, \ldots, T$	$sk_{i+1} \leftarrow (sk^{i+1}, \ldots, sk^T)$	$\sigma \xleftarrow{\$} \mathsf{Sign}'(sk^i, M)$	$d \leftarrow \mathsf{Ver}(pk^i, \sigma, M)$
$\quad (pk^i, sk^i) \xleftarrow{\$} \mathsf{KG}'(par)$	return sk_{i+1}	return (σ, i)	return d
$pk \leftarrow (pk^1, \ldots, pk^T)$			
$sk_1 \leftarrow (sk^1, \ldots, sk^T)$			
return (pk, sk_1)			

Figure 8.2: Naive construction of a forward-secure signature scheme from a standard signature scheme

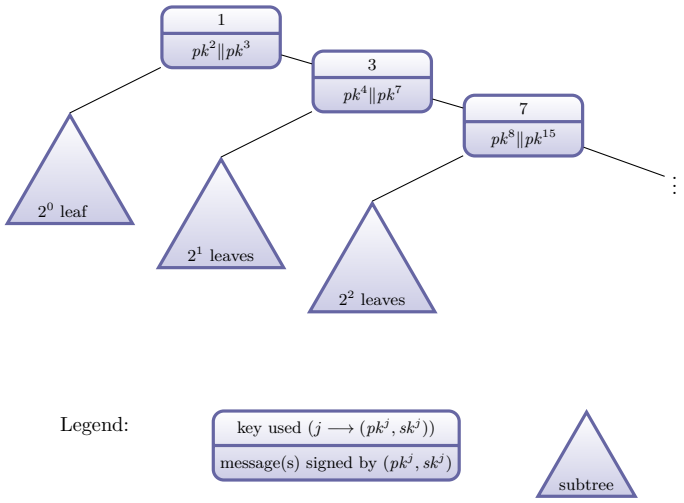

Figure 8.3: Unbalanced certification tree (for an unbounded number of periods)

in a straightforward way. Thus, these two constructions can be used to construct tightly forward-secure signature schemes from tight M-SUF-CMA signature schemes.

This is not directly the case of the constructions in [Kra00] and in [MMM02], due to the use of (forward-secure) pseudorandom generators (PRGs) to generate the randomness for key generation. However, if these PRGs can be modeled by random oracles, then the security of these two constructions can also be tightly reduced to the M-SUF-CMA security of $\mathcal{DS}'$. Furthermore, we remark that the idea of using unbalanced trees in [MMM02] can also be combined with the binary certification tree construction in [BM99], using a binary certification tree as depicted in Figure 8.3. This enables to get a forward-secure scheme for an unbounded number of periods ($T = 2^k$), while still having verification time and signature size for period i only linear in $\log i$ (instead of linear in $\log T = k$).

We refer the reader to [MMM02] for comparison of these various constructions.

9 Constructions of Tightly Secure M-SUF-CMA Signature Schemes

In this section, after recalling some basic tools in Section 9.1, we provide two constructions of M-SUF-CMA signature schemes with tight security reductions to their underlying hard problem, one based on simulation-extractable non-interactive zero-knowledge proofs and another one based on one-time M-SUF-CMA schemes.

As tight M-SUF-CMA signature schemes directly yield tight forward-secure signature schemes, we need to get around the impossibility result from Section 7. Hence, our schemes need to be non-key-verifiable. Intuitively, in all our constructions, the idea is to have at least two possible (perfectly indistinguishable) secret keys for the same public key, such that knowing one secret key and a signature for another secret

key can be used to solve some hard problem. Therefore, the reduction consists in generating honestly all the public and secret keys and hoping that the adversary uses another secret key for its forgery. Since it cannot know which secret key is used by the reduction, this will happen with probability at least $1/2$, independently of the number of issued signatures or of the number of issued public keys (or periods, when the scheme is transformed into a forward-secure signature scheme).

We would like to remark that, despite having tight security reductions, the signature schemes in this section are mostly of theoretical interest since they are significantly less efficient than the non-tight schemes in previous sections.

9.1 Preliminaries

Collision-Resistant Hash Functions

A hash function family $(\mathcal{H}_k)_k$ is a family of functions $\mathfrak{H}$ from $\{0,1\}^*$ to a fixed-length output, namely $\mathbb{Z}_p$ in this paper (with p a prime number). Such a family is said (t,ε)-**collision-resistant** if any adversary $\mathcal{A}$ running in time at most t cannot find a collision with probability more than ε:

$$\Pr\left[M_0 \neq M_1 \text{ and } \mathfrak{H}(M_0) = \mathfrak{H}(M_1) \;\middle|\; \mathfrak{H} \xleftarrow{\$} \mathcal{H}_k \,;\, (M_0, M_1) \xleftarrow{\$} \mathcal{A}(\mathfrak{H}) \right] \leq \varepsilon.$$

Discrete Logarithm

Let us denote by $(p, \mathbb{G}, g)$ a cyclic group of prime order p generated by g. Let us recall a classical problem: discrete logarithm.

Definition 9.1 [Discrete Logarithm Problem (DL)] The Discrete Logarithm assumption says that, in a group $(p, \mathbb{G}, g)$, when we are given g^x for unknown random $x \xleftarrow{\$} \mathbb{Z}_p$, it is hard to find x. More precisely, the DL problem is (t,ε)-hard if no adversary running in time t, can compute x from g^x with probability at most ε.

Commitment Scheme

A **commitment scheme** allows a user to commit to a value without revealing it but without being able to later change his mind. In this paper, we only consider perfectly hiding commitment schemes. More formally, a commitment is defined by a tuple $\mathcal{C} = (\mathcal{C}.\mathsf{Setup}, \mathsf{Commit}, \mathsf{Ver})$ such that:

- $\mathcal{C}.\mathsf{Setup}$ is a probabilistic polynomial time algorithm which takes as input a unary representation of the security parameter k and outputs a common reference string (CRS) crs;
- Commit is a probabilistic polynomial time algorithm which takes as input the CRS crs, an element X from some set $\mathcal{X}_\mathcal{C}$, and outputs a bit string c, called a commitment c to X, and a bit string δ, called a decommitment;
- Ver is a probabilistic algorithm which takes as input the CRS crs, a commitment c, a corresponding decommitment δ, and the committed element X; and outputs 1 to indicate acceptance and 0 otherwise;

and such that it satisfies the three following properties:

1. **Correctness.** A commitment scheme is correct if a commitment and decommitment generated honestly are correctly verified. More formally, for any security parameter k, for any $crs \xleftarrow{\$} \mathcal{C}.\mathsf{Setup}(1^k)$, and for all $X \in \mathcal{X}_\mathcal{C}$, we have:

$$\Pr\left[\mathsf{Ver}(crs, c, \delta, X) = 1 \;\middle|\; (c,\delta) \xleftarrow{\$} \mathsf{Commit}(crs, X) \right] = 1;$$

2. **Perfectly hiding.** A commitment scheme is perfectly hiding if an adversary, even powerful, cannot know which message is committed in a commitment c. More formally, $\mathcal{C}$ is said perfectly hiding, if the distributions of commitments c to X are identical, for all $X \in \mathcal{X}_{\mathcal{C}}$;

3. **Binding.** A commitment scheme is binding if an adversary cannot produce a commitment and two decommitments for two different messages. More formally, $\mathcal{C}$ is said (t, ε)-binding, if for any adversary $\mathcal{A}$ running in time at most t:

$$\Pr\Big[X_0 \neq X_1,\ \mathsf{Ver}(crs, c, \delta_0, X_0) = 1, \text{ and } \mathsf{Ver}(crs, c, \delta_1, X_1) = 1 \mid crs \xleftarrow{\$} \mathcal{C}.\mathsf{Setup}(1^k)\ ;\ (c, \delta_0, X_0, \delta_1, X_1) \xleftarrow{\$} \mathcal{A}(crs) \Big] \leq \varepsilon.$$

For our constructions, we suppose that we can sample a uniform value X from $\mathcal{X}_{\mathcal{C}}$ and that the cardinal $|\mathcal{X}_{\mathcal{C}}|$ is superpolynomial in the security parameter k.

Simulation-Extractable Non-Interactive Zero-Knowledge Proofs

Let us first recall the notion of (labeled) simulation-extractable non-interactive zero-knowledge proof. We consider the quasi-adaptive setting, where the common reference string is allowed to depend on the language [JR13].

Non-interactive proof systems. Intuitively a proof system is a protocol which enables a prover to prove to a verifier that a given word or statement x is in a given NP-language. We are interested in non-interactive proofs, i.e., proofs such that the prover just sends one message.

More formally, let $(\mathcal{L}_{\mathsf{lpar}})_{\mathsf{lpar}}$ be a family of languages in NP (indexed by some parameter lpar) with witness relation $\mathcal{R}_{\mathsf{lpar}}$, i.e., $\mathcal{L}_{\mathsf{lpar}} = \{x \mid \exists \omega,\ \mathcal{R}_{\mathsf{lpar}}(x, \omega) = 1\}$. We suppose lpar is generated by a probabilistic polynomial time algorithm $\mathsf{L.Setup}$ taking as input the a unary representation of the security parameter. Furthermore, we suppose that $\mathcal{R}_{\mathsf{lpar}}(x, \omega)$ can be checked in polynomial time in the security parameter. In the sequel, we often omit lpar when it is clear from context.

A labeled non-interactive proof system for $\mathcal{L}$ is defined by a tuple $\Pi = (\Pi.\mathsf{Setup}, \mathsf{Prove}, \mathsf{Ver})$, such that:

- $\Pi.\mathsf{Setup}$ is a probabilistic polynomial time algorithm which takes as inputs a unary representation of the security parameter k and a language parameter lpar, and outputs a common reference string (CRS) crs;
- Prove is a probabilistic polynomial time algorithm which takes as input the CRS crs, a label $\ell \in \{0,1\}^*$, a word $x \in \mathcal{L}$, and a witness ω for x (such that $\mathcal{R}(x, \omega) = 1$), and outputs a proof π with label ℓ that x is in $\mathcal{L}$;
- Ver is a deterministic algorithm which takes as input the CRS crs, a label $\ell \in \{0,1\}^*$, a word x, and a proof π and outputs 1 to indicate acceptance and 0 otherwise;

and such that it verifies the two following properties:

1. **Perfect completeness.** A non-interactive proof is complete if an honest prover knowing a statement $x \in \mathcal{L}$ and a witness ω for x can convince an honest verifier that x is in $\mathcal{L}$, for any label. More formally, Π is said perfectly complete, if for any security parameter k, for any $\ell \in \{0,1\}^*$, for any $\mathsf{lpar} \xleftarrow{\$} \mathsf{L.Setup}(1^k)$, for any $x \in \mathcal{L}$ and ω such that $\mathcal{R}(x, \omega) = 1$, for any $crs \xleftarrow{\$} \Pi.\mathsf{Setup}(1^k)$, we have $\mathsf{Ver}(crs, \ell, x, \mathsf{Prove}(crs, \ell, x, \omega)) = 1$;

2. **Soundness.** A non-interactive proof is said (quasi-adaptively) sound, if no polynomial time adversary $\mathcal{A}$ can prove a false statement with non-negligible probability. More formally, Π is (t, ε)-sound if for any adversary running in time at most t and any $\mathsf{lpar} \xleftarrow{\$} \mathsf{L.Setup}(1^k)$:

$$\Pr\Big[\mathsf{Ver}(crs, \ell, x, \pi) = 1 \text{ and } x \notin \mathcal{L} \ \Big|\ crs \xleftarrow{\$} \Pi.\mathsf{Setup}(1^k, \mathsf{lpar})\ ;\ (\ell, x, \pi) \xleftarrow{\$} \mathcal{A}(crs) \Big] \leq \varepsilon.$$

Non-interactive zero-knowledge proofs (NIZK). An (unbounded) **NIZK** (non-interactive zero-knowledge proof) is a non-interactive proof system with two simulators Sim_1 and Sim_2, which can simulate $\Pi.\mathsf{Setup}$ and Prove, but such that Sim_2 does not need any witness. More formally a NIZK is defined by a tuple $\Pi = (\Pi.\mathsf{Setup}, \mathsf{Prove}, \mathsf{Ver}, \mathsf{Sim}_1, \mathsf{Sim}_2)$ such that $(\Pi.\mathsf{Setup}, \mathsf{Prove}, \mathsf{Ver})$ is a non-interactive proof system, and:

- Sim_1 is a probabilistic algorithm which takes as inputs a unary representation of k and a language parameter lpar, and generates a CRS crs and a trapdoor τ, such that Sim_2 can use τ to simulate proofs under crs;
- Sim_2 is a probabilistic algorithm which takes as input the CRS crs, a corresponding trapdoor τ, a label ℓ, a word x (not necessarily in $\mathcal{L}$), and outputs a (fake or simulated) proof π for x;

and such that it satisfies the following property:

- **Unbounded zero-knowledge.** A NIZK is said (unbounded) zero-knowledge if simulated proofs are indistinguishable from real proofs. More formally, Π is (t, ε)-unbounded-zero-knowledge if, for any adversary running in time at most t and any $\mathsf{lpar} \xleftarrow{\$} \mathsf{L.Setup}(1^k)$:

$$\Big|\Pr\Big[\mathcal{A}(crs)^{\mathsf{Prove}(crs,\cdot,\cdot,\cdot)} = 1 \;\Big|\; crs \xleftarrow{\$} \Pi.\mathsf{Setup}(1^k, \mathsf{lpar})\Big] - \Pr\Big[\mathcal{A}(crs)^{\mathsf{Sim}'(crs,\tau,\cdot,\cdot,\cdot)} = 1 \;\Big|\; (crs,\tau) \xleftarrow{\$} \mathsf{Sim}_1(1^k, \mathsf{lpar})\Big]\Big| \leq \varepsilon$$

  where $\mathsf{Sim}'(crs, \tau, \ell, x, \omega) = \mathsf{Sim}_2(crs, \tau, \ell, x)$ if $\mathcal{R}(x, \omega) = 1$ and $\perp$ otherwise.

We are also interested in a stronger property than soundness:

- **Simulation extractability.** A NIZK is said simulation-extractable if there exists a polynomial time algorithm Ext which can extract a witness from any proof generated by the adversary, even if the adversary can see simulated proofs. More formally, Π is (t, ε)-simulation-sound-extractable if, for any adversary running in time at most t and any $\mathsf{lpar} \xleftarrow{\$} \mathsf{L.Setup}(1^k)$:

$$\Pr\Big[\mathsf{Ver}(crs, x, \pi) = 1,\ (\ell, x, \pi) \notin S, \text{ and } \mathcal{R}(x, \mathsf{Ext}(crs, \tau, \ell, x, \pi)) = 0 \mid (crs,\tau) \xleftarrow{\$} \mathsf{Sim}_1(1^k, \mathsf{lpar})\,;\ (x,\pi) \xleftarrow{\$} \mathcal{A}^{\mathsf{Sim}_2(crs,\tau,\cdot,\cdot)}(crs)\Big] \leq \varepsilon$$

  where S is the set of queries-answers (ℓ, x, π) from Sim_2.

We call a simulation-extractable NIZK, an SE-NIZK.

9.2 Construction Based on Commitments and Simulation-Extractable NIZKs

Generic construction. We construct a M-SUF-CMA signature scheme from a (perfectly hiding) commitment scheme and a simulation-extractable NIZK. The construction is depicted in Figure 9.1. The public key is a commitment c of a random value $X \xleftarrow{\$} \mathcal{X}_{\mathcal{C}}$. Then a signature of a message M is an SE-NIZK, labeled by the message M, which proves the knowledge of the committed value X and the associated decommitment information δ. The verification consists in checking the SE-NIZK is correct and labeled with M.

We remark that this construction is very similar to the leakage-resilient signature scheme of Haralambiev in [Har11], which is a variant of the scheme of Katz and Vaikuntanathan [KV09].

Security. We can tightly reduce the M-SUF-CMA security to the binding property of the commitment scheme and to the unbounded zero-knowledge and simulation extractability properties of the SE-NIZK.

$\underline{\mathsf{PG}(1^k)}$	$\underline{\mathsf{KG}(par, 1^k)}$	$\underline{\mathsf{Sign}(par, sk, M)}$	$\underline{\mathsf{Ver}(par, pk, \sigma, M)}$
$crs_{\mathcal{C}} \xleftarrow{\$} \mathcal{C}.\mathsf{Setup}(1^k)$	$(crs_{\mathcal{C}}, crs_{\Pi}) \leftarrow par$	$(crs_{\mathcal{C}}, crs_{\Pi}) \leftarrow par$	$(crs_{\mathcal{C}}, crs_{\Pi}) \leftarrow par$
$\mathsf{lpar} \leftarrow crs_{\mathcal{C}}$	$X \xleftarrow{\$} \mathcal{X}_{\mathcal{C}}$	$(X, \delta) \leftarrow sk$	$c \leftarrow pk$; $\pi \leftarrow \sigma$
$crs_{\Pi} \xleftarrow{\$} \Pi.\mathsf{Setup}(1^k, \mathsf{lpar})$	$(c, \delta) \xleftarrow{\$} \mathsf{Commit}(crs_{\mathcal{C}}, X)$	$\pi \xleftarrow{\$} \mathsf{Prove}(crs_{\Pi}, M, c, (X, \delta))$	return $\mathsf{Ver}(crs_{\Pi}, M, c, \pi)$
$par \leftarrow (crs_{\mathcal{C}}, crs_{\Pi})$	$pk \leftarrow c$; $sk \leftarrow (X, \delta)$	$\sigma \leftarrow \pi$	
return par	return (pk, sk)	return σ	

Figure 9.1: A M-SUF-CMA signature scheme from a commitment scheme $\mathcal{C}$ and a SE-NIZK Π for the language $\mathcal{L}_{\mathcal{C}_{crs}} = \{c \mid \exists (X, \delta), \mathsf{Ver}(crs_{\mathcal{C}}, c, \delta, X) = 1\}$

Intuitively, this comes from the fact the adversary does not know which X the signer (or the reduction) has chosen. Therefore, if it forges a signature for some public key $pk = c$, then the reduction can extract a decommitment information δ and a message X different from the one used to create $pk = c$. And this is computationally hard, due to the binding property of the commitment scheme.

Formally, we have the following security theorem.

Theorem 9.2 *The signature scheme depicted in Figure 9.1 is M-SUF-CMA. More precisely, if the underlying commitment scheme $\mathcal{C}$ is $(t_{\mathcal{C}}, \varepsilon_{\mathcal{C}})$-binding and if the underlying SE-NIZK Π is (t_z, ε_z)-unbounded-zero-knowledge, (t_e, ε_e)-simulation-extractable, then the signature schemes is (t, ε)-forward-secure for:*

$$\varepsilon = \varepsilon_z + \varepsilon_e + \frac{1}{|\mathcal{X}_{\mathcal{C}}|} + \varepsilon_{\mathcal{C}} \qquad \text{and} \qquad t \approx \min(t_z, t_e, t_{\mathcal{C}}),$$

where $|\mathcal{X}_{\mathcal{C}}|$ is the cardinal of $\mathcal{X}_{\mathcal{C}}$, t_{KG} denotes the average time of an execution of KG, t_{Sign} denotes the average time of a query to Sign, t_{exp} is the time for an exponentiation in the cyclic group $\mathbb{G}$, and q_s denotes the total number of signature queries.

Proof: Let us just sketch the games of the proof here.

Game $\mathbf{G}_0$: we simulate all SE-NIZK. This game is indistinguishable from the original game due to the unbounded zero-knowledge property of the SE-NIZK.

Game $\mathbf{G}_1$: we extract the witness of the proof of the forged signature σ^* (for message M^* and public key pk^*) and check that this witness is a pair (X^*, δ^*) such that $\mathsf{Ver}(crs_{\mathcal{C}}, c, \delta^*, X^*)$. If not, then we abort. The probability of aborting is at most ε_e due to the simulation-extractability property of the SE-NIZK.

Game $\mathbf{G}_2$: let X and δ be the message and the corresponding decommitment in sk^* (i.e., the ones used to generate c in pk^*). If $X = X^*$, we abort. This happens with probability at most $1/|\mathcal{X}_{\mathcal{C}}|$, since the adversary has no information on X (nothing he sees depends on this value, as the commitment scheme is perfectly hiding).

Then, for the last game, if the adversary wins, we have opened the commitment c with two different messages: X and X^*, which is computationally hard because of the binding property of the commitment scheme. ∎

Instantiations. For our whole construction to be tight, we need a commitment scheme and an associated SE-NIZK with a tight reduction for the unbounded zero-knowledge and simulation extractability properties, where "tight" means that the reduction does not lose a factor which depends on the number of queries to the oracles Prove, Sim', and Sim_2.

Construction without random oracles. For that purpose, we can use a labeled version of the SE-NIZK1 construction in [ADK+13] (which is a more efficient variant of the SE-NIZK in [HJ12]), simply

$\underline{\mathsf{PG}(1^k)}$	$\underline{\mathsf{KG}(par)}$	$\underline{\mathsf{Sign}(par, sk, M)}$	$\underline{\mathsf{Ver}(par, pk, \sigma, M)}$
Generate $(p, \mathbb{G}, g)$ for k	$b \xleftarrow{\$} \{0,1\}$	$(b, x_b, y_0, y_1) \leftarrow sk$	$(X_0, X_1, Y) \leftarrow pk$
$\mathfrak{H} \xleftarrow{\$} \mathcal{H}_k$	$x_b \xleftarrow{\$} \mathbb{Z}_p^*$; $X_b \leftarrow g^{x_b}$	$s_b \leftarrow y_b - \mathfrak{H}(M)/x_b$	$(s_0, s_1) \leftarrow \sigma$
$par \leftarrow (p, \mathbb{G}, g, \mathfrak{H})$	$X_{1-b} \xleftarrow{\$} \mathbb{G}^*$	$s_{1-b} \leftarrow y_{1-b}$	$Z \leftarrow g^{\mathfrak{H}(M)} \cdot X_0^{s_0} \cdot X_1^{s_1}$
return par	$y_0, y_1 \xleftarrow{\$} \mathbb{Z}_p$; $Y \leftarrow X_0^{y_0} \cdot X_1^{y_1}$	$\sigma \leftarrow (s_0, s_1)$	if $Z = Y$ then
	$pk \leftarrow (X_0, X_1, Y)$	return σ	return 1
	$sk \leftarrow (b, x_b, y_0, y_1)$		else
	return (pk, sk)		return 0

Figure 9.2: A one-time M-SUF-CMA signature scheme

by adding the label to the part signed by the one-time signature. As the original construction, this labeled variant can be proved simulation-extractable under the Decisional Linear (DLin) assumption [BBS04], with a tight reduction.

This SE-NIZK can handle pairing product equations whose right-hand side is a product of pairings of constant group elements, as Groth-Sahai NIZK [GS08]. Therefore, we need a commitment scheme for which messages X and decommitment information δ are group elements, and for which the decommitment algorithm consists in verifying such pairing-product equations. This is the case of the commitment scheme TC3 in [Har11], which is perfectly hiding and computationally binding under the DLin assumption.

Construction with random oracles. In the random-oracle model, we can replace the complex Groth-Sahai-based SE-NIZK with an SE-NIZK based on Ω-protocols [GMY06] and the Fiat-Shamir transform.

9.3 Construction Based on One-Time M-SUF-CMA Schemes

In this section, we show how to build a tightly secure M-SUF-CMA signature scheme based on a one-time M-SUF-CMA signature scheme, where the latter is a M-SUF-CMA signature scheme for which one can sign at most $q_s = 1$ message with respect to each public key. Towards this goal, we first present in Section 9.3 an efficient construction of a one-time M-SUF-CMA signature scheme based on the strong one-time signature scheme proposed by Groth [Gro06]. Interestingly, we remark that one-time M-SUF-CMA signature schemes can already be directly used to build a special type of key-evolving signatures, known as fine-grained forward-secure [CK06], where the signer can sign at most one message in each time period and has to update his or her secret key after each signature.

Next, in Section 9.3, we show how to convert one-time M-SUF-CMA signature scheme into a standard M-SUF-CMA signature scheme with the help of a random oracle. This is achieved by showing that random oracles can help us replace standard M-SUF-CMA signature schemes with their one-time versions for all the internal nodes of the generic forward-secure construction in Section 8.2.

One-time M-SUF-CMA scheme

The scheme we propose is very close to the strong one-time signature scheme proposed by Groth in [Gro06]. It is depicted in Figure 9.2.

Proposition 9.3 *The scheme $\mathcal{DS}$ described above is $(t, I, q_s, 0, \varepsilon)$-one-time-*M-SUF-CMA*, if the DL problem is (t', ε')-hard and $\mathcal{H}$ is (t'', ε'')-collision-resistant (see Section 9.1), for*

$$\varepsilon = \varepsilon'' + 2\varepsilon' \qquad\qquad t \approx \min(t', t'')$$

Before giving a formal proof, let us first sketch the three main ideas of the proof. First, the DL problem is random self-reducible. Second, if the adversary asks for a signature (s_0, s_1) on a message M

Finalize(pk^*, M^*, σ^*) Game G_0, $\boxed{G_1}$

001 $d \leftarrow \mathsf{Ver}(par, pk^*, \sigma^*, M^*)$
002 $sk^* \leftarrow P[pk^*]$
003 if $sk^* = \perp$ then return 0
004 $(X_0, X_1, Y) \leftarrow pk^*$; $(b, x_b, y_0, y_1) \leftarrow sk^*$
005 $(s_0^*, s_1^*) \leftarrow \sigma$
006 $Z \leftarrow g^{\mathfrak{H}(M^*)} \cdot X_0^{s_0^*} \cdot X_1^{s_1^*}$
007 if $Z \neq Y$ then $d \leftarrow 0$
008 if $(pk^*, M^*, \sigma^*) \in S$ then $d \leftarrow 0$
009 find a triple $(M, \sigma, i^*) \in S$
010 if such a triple exists (it is then unique) then
011 $\quad (s_0, s_1) \leftarrow \sigma$
012 else
013 $\quad M \xleftarrow{\$} \mathcal{M} \setminus \{M^*\}$
014 $\quad (s_0, s_1) \leftarrow \mathsf{Sign}(sk, M)$
015 if $M \neq M^*$ and $\mathfrak{H}(M) = \mathfrak{H}(M^*)$
016 $\quad \mathsf{bad} \leftarrow \mathsf{true}$
017 $\quad \boxed{d \leftarrow 0}$
018 return d

Finalize(pk^*, M^*, σ^*) Game G_2, $\boxed{G_3}$

201 $d \leftarrow \mathsf{Ver}(par, pk^*, \sigma^*, M^*)$
202 $sk^* \leftarrow P[pk^*]$
203 if $sk^* = \perp$ then return 0
204 $(X_0, X_1, Y) \leftarrow pk^*$; $(b, x_b, y_0, y_1) \leftarrow sk^*$
205 $(s_0^*, s_1^*) \leftarrow \sigma$
206 $Z \leftarrow g^{\mathfrak{H}(M^*)} \cdot X_0^{s_0^*} \cdot X_1^{s_1^*}$
207 if $Z \neq Y$ then $d \leftarrow 0$
208 if $(pk^*, M^*, \sigma^*) \in S$ then $d \leftarrow 0$
209 find a triple $(M, \sigma, i^*) \in S$
210 if such a triple exists (it is then unique) then
211 $\quad (s_0, s_1) \leftarrow \sigma$
212 else
213 $\quad M \xleftarrow{\$} \mathcal{M} \setminus \{M^*\}$
214 $\quad (s_0, s_1) \leftarrow \mathsf{Sign}(sk, M)$
215 if $M \neq M^*$ and $\mathfrak{H}(M) = \mathfrak{H}(M^*)$ then $d \leftarrow 0$
216 if $d = 1$ and $s_{1-b_i} = s_{1-b_i}^*$
217 $\quad \mathsf{bad} \leftarrow \mathsf{true}$
218 $\quad \boxed{d \leftarrow 0}$
219 return d

Figure 9.3: Games $G_0, \ldots, G_3$ for proof of Proposition 9.3. G_1 includes the boxed code at line 017 but G_0 does not. G_3 includes the boxed code at line 218 but G_2 does not.

and produces a signature (s_0^*, s_1^*) on a message M^* such that $s_{1-b}^* \neq s_{1-b}$, one can compute the discrete logarithm x_{1-b} of X_{1-b}:

$$x_{1-b} = \frac{\mathfrak{H}(M) + x_b s_b - \mathfrak{H}(M^*) - x_b s_b^*}{s_{1-b}^* - s_{1-b}}.$$

Third, the bit b cannot be known by the adversary (it is completely independent from (s_0, s_1) and the public key pk^*), and so a valid forgery of the adversary will satisfy the above property with probability at least $1/2$ (if we ignore collisions in the hash function).

Proof: Suppose there exists an adversary $\mathcal{A}$ which $(t, I, q_s, 0, \varepsilon)$-breaks the one-time M-SUF-CMA-security of $\mathcal{DS}$. Let us consider the games $G_0, \ldots, G_3$ of Figure 9.3.

G_0 corresponds to a slight variant of the one-time version of the game $\mathbf{Exp}^{\text{m-suf-cma}}_{\mathcal{DS},k}(\mathcal{A})$ defining the one-time M-SUF-CMA-security of $\mathcal{DS}$. Only the **Finalize** procedure is depicted in Figure 9.3, the other procedures are the same as in $\mathbf{Exp}^{\text{m-suf-cma}}_{\mathcal{DS},k}(\mathcal{A})$. Furthermore we set bad when the adversary submit a message M^* for a public key pk^* which has the same hash as the message M it queried to the signing oracle **Sign** for public key pk^* (or a random message M if it has not queried the signing oracle for public key pk_i^*). G_0 has the same output as the original game.

Since, when bad is set, there is a collision in the hash function ($\mathfrak{H}(M) = \mathfrak{H}(M^*)$ but $M \neq M^*$), $\Pr[\, G_0(\mathcal{A}) \text{ sets } \mathsf{bad} \,] \leq \varepsilon''$. In G_1, when bad is set, **Finalize** rejects the forged signature and outputs 0. Since G_0 and G_1 are identical until bad, thanks to Lemma 2.1, we have

$$\Pr[\, G_0(\mathcal{A}) \Rightarrow 1 \,] - \Pr[\, G_1(\mathcal{A}) \Rightarrow 1 \,] \leq \Pr[\, G_0(\mathcal{A}) \text{ sets } \mathsf{bad} \,] \leq \varepsilon''.$$

In G_2, bad is now set when $s_{1-b} = s_{1-b}^*$. These two modifications do not change the output of the game and so $\Pr[\, G_1(\mathcal{A}) \Rightarrow 1 \,] = \Pr[\, G_2(\mathcal{A}) \Rightarrow 1 \,]$.

Let us now prove that $\Pr[\, G_2(\mathcal{A}) \text{ sets } \mathsf{bad} \,] \leq \Pr[\, G_2(\mathcal{A}) \Rightarrow 1 \,]/2$. Let us suppose $G_2(\mathcal{A}) \Rightarrow 1$. We have the following equation:

$$Y = g^{\mathfrak{H}(M)} \cdot X_0^{s_0} \cdot X_1^{s_1} = g^{\mathfrak{H}(M^*)} \cdot X_0^{s_0^*} \cdot X_1^{s_1^*}$$

which implies

$$\mathfrak{H}(M) + s_0\, x_0 + s_1\, x_1 = \mathfrak{H}(M^*) + s_0^*\, x_0 + s_1^*\, x_1. \tag{9.1}$$

If $M = M^*$, since $(pk^*, M^*, \sigma^*) \notin S$ (otherwise $d = 0$), then $(s_0, s_1) \neq (s_0^*, s_1^*)$. Otherwise, $M = M^*$, and thus $\mathfrak{H}(M) \neq \mathfrak{H}(M')$, and we also have $(s_0, s_1) \neq (s_0^*, s_1^*)$. Let us suppose $s_0 \neq s_0^*$ without loss of generality (the proof works similarly when $s_1 \neq s_1^*$).

Let us now show that conditioned on the view of the adversary, $b = 0$ with probability $\frac{1}{2}$. For that, we remark that knowing all the public keys pk and their associated secret keys sk except sk^* and knowing M, M^*, s_0, s_1, for each value of b (0 or 1), there exist exactly one unique corresponding value for the pair (y_0, y_1):

$$y_{1-b} = s_{1-b} \qquad\qquad y_b = s_b + \mathfrak{H}(M)/x_b,$$

where $x_0, x_1 \in \mathbb{Z}_p$ are defined by $X_0 = g^{x_0}$ and $X_1 = g^{x_1}$. Therefore $s_{1-b} \neq s_{1-b}^*$ with this probability, namely $\frac{1}{2}$, and $\Pr[\,\mathrm{G}_2(\mathcal{A}) \text{ sets } \mathsf{bad}\,] \leq \Pr[\,\mathrm{G}_2(\mathcal{A}) \Rightarrow 1\,]/2$.

In G_3, when bad is set, **Finalize** rejects the forged signature and outputs 0. Since G_2 and G_3 are identical until bad, thanks to Lemma 2.1, we have

$$\Pr[\,\mathrm{G}_2(\mathcal{A}) \Rightarrow 1\,] - \Pr[\,\mathrm{G}_3(\mathcal{A}) \Rightarrow 1\,] \leq \Pr[\,\mathrm{G}_2(\mathcal{A}) \text{ sets } \mathsf{bad}\,] \leq \Pr[\,\mathrm{G}_2(\mathcal{A}) \Rightarrow 1\,]/2.$$

and so $\Pr[\,\mathrm{G}_2(\mathcal{A}) \Rightarrow 1\,] \leq 2\Pr[\,\mathrm{G}_3(\mathcal{A}) \Rightarrow 1\,]$.

Now let us prove that $\Pr[\,\mathrm{G}_3(\mathcal{A}) \Rightarrow 1\,] \leq \varepsilon'$. Indeed, from $\mathcal{A}$, we can create an adversary which can compute the DL of any element $X \in \mathbb{G}^*$. We just need to simulate the game G_3, except we compute X_{1-b} as X^r for a random $r \in \mathbb{Z}_p^*$ (instead of picking it at random in $\mathbb{G}^*$) for all keys generated by **KG** (not only pk^*). Then, if $\mathcal{A}$ wins the game, we can easily compute the discrete logarithm x of X because, according to Equation (9.1):

$$r\, x = x_{1-b} = \frac{\mathfrak{H}(M) + x_b s_b - \mathfrak{H}(M^*) - x_b s_b^*}{\mathrm{forge} s_{1-b} - s_{1-b}}.$$

Therefore $\Pr[\,\mathrm{G}_3(\mathcal{A}) \Rightarrow 1\,] \leq \varepsilon'$.

From the previous equalities and inequalities, we deduce that, for any adversary $\mathcal{A}$ running in time approximately at most t, its probability success is $\varepsilon \leq \Pr[\,\mathrm{G}_0(\mathcal{A}) \Rightarrow 1\,] \leq \varepsilon'' + 2\varepsilon'$. ∎

From One-Time M-SUF-CMA to M-SUF-CMA

In this section, we describe a tightness-preserving transform from a one-time M-SUF-CMA scheme to a (standard) M-SUF-CMA scheme, using a hash function modeled as a random oracle. A similar idea was used by Goldreich in [Gol87] to render the GMR signature scheme [GMR84] memoryless.

The main idea behind the construction is to implicitly construct a certification tree, as depicted in Figure 9.4. In this tree, each node j is associated with a fresh pair of public and secret keys for the underlying one-time M-SUF-CMA scheme, where each internal node's secret key is used to sign the public keys of its children and where the secret keys associated with the leaves are used to sign the actual messages. In order to avoid having to store the entire tree or to maintain a state, the randomness used by the key-generation and signing algorithms of each node are computed in a deterministic manner via a hash function, which is modeled as a random oracle, using a random seed and the node position as input. Moreover, in order to avoid reusing the same leaf twice for signing two different messages, the choice of the leaf used to sign a message is also done via a hash function, using the same random seed and the message itself as input.

More precisely, let $\mathcal{DS} = (\mathsf{PG}, \mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be a one-time M-SUF-CMA scheme and let H_1 and H_2 be two hash functions, modeled as random oracles, which:

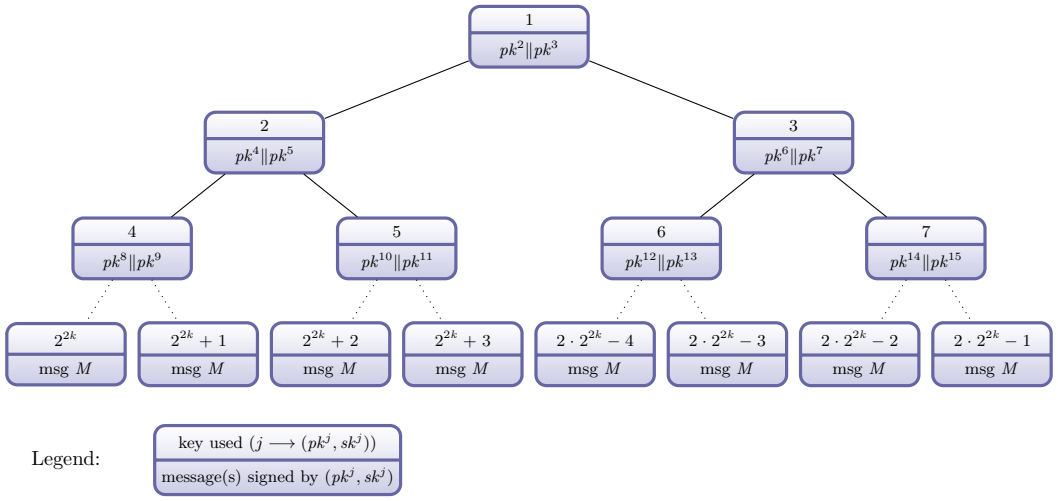

Figure 9.4: Certification tree

- on input $s\|j$ ($s \in \{0,\ldots,2^{2k}-1\}, j \in \{1,\ldots,2^{2k}\}$), H_1 outputs a pair $(r_{\mathsf{KG}}, r_{\mathsf{Sign}})$ of a random tape r_{KG} for KG and a random tape r_{Sign} for Sign;
- on input $s\|M$ ($s \in \{0,\ldots,2^{2k}-1\}, M \in \mathcal{M}$), H_2 outputs an integer i in $\{1,\ldots,2^{2k}\}$.

We then construct a classical M-SUF-CMA scheme $\mathcal{DS}' = (\mathsf{PG}', \mathsf{KG}', \mathsf{Sign}', \mathsf{Ver}')$ as follows. The common parameter generation algorithm PG', on input 1^k, simply runs $par \xleftarrow{\$} \mathsf{PG}(1^k)$ and outputs par. The key generation KG', on input $(par, 1^k)$, starts by choosing a random seed $s \in \{0,\ldots,2^{2k}-1\}$, which will play the role of the secret signing key. It then computes $(r_{\mathsf{KG}}, r_{\mathsf{Sign}}) = \mathsf{H}_1(s\|1)$ followed by $(sk_1, pk_1) = \mathsf{KG}(par, 1^k; r_{\mathsf{KG}})$ and outputs (pk_1, s) as its public and secret keys. The signing algorithm Sign', on input (par, s, M), selects the leaf labeled $2^{2k} - 1 + \mathsf{H}_2(s\|M)$, computes all the signatures along the path from the root of the certification tree to this leaf, using $(r_{\mathsf{KG}_j}, r_{\mathsf{Sign}_j}) = \mathsf{H}_1(s\|j)$ as the randomness for the key-generation and signing algorithms for each node j in the path. It then outputs this list of $2k+1$ signatures together with the corresponding public keys as the signature σ of M. The verification algorithm Ver, on input (par, pk_1, σ, M) simply checks that all the signatures in the list are valid using the corresponding public keys.

If each message is signed using a different leaf (which happens with overwhelming probability), each key sk_j is used to sign only one message. It is then easy to see that the M-SUF-CMA security of this new signature scheme can be tightly reduced to the one-time M-SUF-CMA of the underlying signature scheme: a forgery to the new scheme directly implies a forgery for the one-time scheme (for at least one of the public keys on the path from the message of the forgery to the root of the tree).

The reader acquainted with the construction of Goldreich may wonder why we cannot simply use a PRF instead of a random oracle, as in the original construction. The reason for this is that, since the key of the PRF would need to be stored in the secret key of the signature scheme, we would need to know in advance which secret keys will be corrupted. While we would need to compute honestly the output of the PRF for corrupted secret keys, we would need to use the pseudorandomness property of the PRF to use random values for the output of the PRF for the secret key corresponding to the forgery.

Acknowledgments

We would like to thank Mihir Bellare and Eike Kiltz for their helpful comments on a preliminary version of this paper, the anonymous referees of PKC 2013 for their valuable input, and Benoît Libert for his discussion with the second author on simulation-sound NIZK and random oracles. We would also like to thank the anonymous reviewers for Journal of Cryptology for their insightful comments.

This work was supported in part by the French ANR-10-SEGI-015 PRINCE Project, in part by the CFM Foundation, and in part by the European Commission through the FP7-ICT-2011-EU-Brazil Program under Contract 288349 SecFuNet and the ICT Program under Contract ICT-2007-216676 ECRYPT II. The second author was supported in part by the Defense Advanced Research Projects Agency (DARPA) and Army Research Office (ARO) under Contract No. W911NF-15-C-0236.

References

[AABN02] M. Abdalla, J. H. An, M. Bellare, and C. Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In *EUROCRYPT 2002*, *LNCS* 2332, pages 418–433. Springer, Heidelberg, April / May 2002. (pages 1 and 12.)

[ABP13] M. Abdalla, F. Ben Hamouda, and D. Pointcheval. Tighter reductions for forward-secure signature schemes. In *PKC 2013*, *LNCS* 7778, pages 292–311. Springer, Heidelberg, February / March 2013. (pages 2 and 3.)

[ADK+13] M. Abe, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo. Tagged one-time signatures: Tight security and optimal tag size. In *PKC 2013*, *LNCS* 7778, pages 312–331. Springer, Heidelberg, February / March 2013. (page 39.)

[AFLT12] M. Abdalla, P.-A. Fouque, V. Lyubashevsky, and M. Tibouchi. Tightly-secure signatures from lossy identification schemes. In *EUROCRYPT 2012*, *LNCS* 7237, pages 572–590. Springer, Heidelberg, April 2012. (pages 1, 8, 11, and 12.)

[And00] R. Anderson. Two remarks on public-key cryptology. Manuscript. Relevant material presented by the author in an invited lecture at the 4th ACM Conference on Computer and Communications Security, CCS 1997, Zurich, Switzerland, April 1–4, 1997, September 2000. (page 34.)

[BBS04] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO 2004*, *LNCS* 3152, pages 41–55. Springer, Heidelberg, August 2004. (page 40.)

[BHJL16] F. Benhamouda, J. Herranz, M. Joye, and B. Libert. Efficient cryptosystems from 2^k-th power residue symbols. *Journal of Cryptology*, 2016. To appear. (pages 2 and 6.)

[BJLS16] C. Bader, T. Jager, Y. Li, and S. Schäge. On the impossibility of tight cryptographic reductions. In *EUROCRYPT 2016, Part II*, *LNCS* 9666, pages 273–304. Springer, Heidelberg, May 2016. (pages 3 and 30.)

[BM99] M. Bellare and S. K. Miner. A forward-secure digital signature scheme. In *CRYPTO'99*, *LNCS* 1666, pages 431–448. Springer, Heidelberg, August 1999. (pages 6, 7, 8, 32, 34, and 35.)

[BMO90] M. Bellare, S. Micali, and R. Ostrovsky. The (true) complexity of statistical zero knowledge. In *22nd ACM STOC*, pages 494–502. ACM Press, May 1990. (page 10.)

[BNN07] M. Bellare, C. Namprempre, and G. Neven. Unrestricted aggregate signatures. In *ICALP 2007*, *LNCS* 4596, pages 411–422. Springer, Heidelberg, July 2007. (page 4.)

[BP97] N. Bari and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In *EUROCRYPT'97*, *LNCS* 1233, pages 480–494. Springer, Heidelberg, May 1997. (page 6.)

[BPS16] M. Bellare, B. Poettering, and D. Stebila. From identification to signatures, tightly: A framework and generic transforms. In *ASIACRYPT 2016, Part II*, *LNCS* 10032, pages 435–464. Springer, Heidelberg, December 2016. (page 1.)

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS 93*, pages 62–73. ACM Press, November 1993. (pages 2 and 4.)

[BR06] M. Bellare and P. Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *EUROCRYPT 2006*, *LNCS* 4004, pages 409–426. Springer, Heidelberg, May / June 2006. (page 4.)

[BS96] E. Bach and J. Shallit. *Algorithmic Number Theory*. MIT Press, August 1996. (pages 5 and 23.)

[CD95] R. Cramer and I. Damgård. Escure signature schemes based on interactive protocols. In *CRYPTO'95*, *LNCS* 963, pages 297–310. Springer, Heidelberg, August 1995. (page 52.)

[CK06] J. Camenisch and M. Koprowski. Fine-grained forward-secure signature schemes without random oracles. *Discrete Applied Mathematics*, 154(2):175–188, 2006. (page 40.)

[CMS99] C. Cachin, S. Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *EUROCRYPT'99*, *LNCS* 1592, pages 402–414. Springer, Heidelberg, May 1999. (pages 1, 2, and 5.)

[Cor02] J.-S. Coron. Optimal security proofs for PSS and other signature schemes. In *EUROCRYPT 2002*, *LNCS* 2332, pages 272–287. Springer, Heidelberg, April / May 2002. (pages 2, 30, 31, and 32.)

[Cra96] R. Cramer. *Modular Design of Secure Yet Practical Cryptographic Protocols*. PhD thesis, CWI and University of Amsterdam, Amsterdam, The Netherlands, November 1996. (page 10.)

[Dus98] P. Dusart. Autour de la fonction qui compte le nombre de nombres premiers. *Thesis, Université de Limoges*, 1998. (page 51.)

[ECR11] ECRYPT II yearly report on algorithms and keysizes, 2011. (page 26.)

[FFS88] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, 1988. (page 1.)

[FO97] E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO'97*, *LNCS* 1294, pages 16–30. Springer, Heidelberg, August 1997. (page 6.)

[FS87] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO'86*, *LNCS* 263, pages 186–194. Springer, Heidelberg, August 1987. (pages 1 and 11.)

[GBL08] S. Garg, R. Bhaskar, and S. V. Lokam. Improved bounds on security reductions for discrete log based signatures. In *CRYPTO 2008*, *LNCS* 5157, pages 93–107. Springer, Heidelberg, August 2008. (page 1.)

[GMR84] S. Goldwasser, S. Micali, and R. L. Rivest. A "paradoxical" solution to the signature problem (extended abstract). In *25th FOCS*, pages 441–448. IEEE Computer Society Press, October 1984. (pages 7 and 42.)

[GMR89] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. (page 10.)

[GMY06] J. A. Garay, P. D. MacKenzie, and K. Yang. Strengthening zero-knowledge protocols using signatures. *Journal of Cryptology*, 19(2):169–209, April 2006. (page 40.)

[Gol87] O. Goldreich. Two remarks concerning the Goldwasser-Micali-Rivest signature scheme. In *CRYPTO'86*, *LNCS* 263, pages 104–110. Springer, Heidelberg, August 1987. (page 42.)

[GQ88] L. C. Guillou and J.-J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both trasmission and memory. In *EUROCRYPT'88*, *LNCS* 330, pages 123–128. Springer, Heidelberg, May 1988. (pages 1, 17, and 18.)

[Gro06] J. Groth. Simulation-sound NIZK proofs for a practical language and constant size group signatures. In *ASIACRYPT 2006*, *LNCS* 4284, pages 444–459. Springer, Heidelberg, December 2006. (page 40.)

[GS08] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, *LNCS* 4965, pages 415–432. Springer, Heidelberg, April 2008. (page 40.)

[Har11] K. Haralambiev. *Efficient Cryptographic Primitives for Non-Interactive Zero-Knowledge Proofs and Applications*. PhD thesis, New York University, 2011. (pages 38 and 40.)

[HJ12] D. Hofheinz and T. Jager. Tightly secure signatures and public-key encryption. In *CRYPTO 2012*, *LNCS* 7417, pages 590–607. Springer, Heidelberg, August 2012. (page 39.)

[HW09] S. Hohenberger and B. Waters. Short and stateless signatures from the RSA assumption. In *CRYPTO 2009*, *LNCS* 5677, pages 654–670. Springer, Heidelberg, August 2009. (page 22.)

[IN96] R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology*, 9(4):199–216, 1996. (page 1.)

[IR01] G. Itkis and L. Reyzin. Forward-secure signatures with optimal signing and verifying. In *CRYPTO 2001*, *LNCS* 2139, pages 332–354. Springer, Heidelberg, August 2001. (pages 2, 3, 5, 6, 17, 22, 23, 25, and 55.)

[JL13] M. Joye and B. Libert. Efficient cryptosystems from 2^k-th power residue symbols. In *EUROCRYPT 2013*, *LNCS* 7881, pages 76–92. Springer, Heidelberg, May 2013. (pages 2 and 6.)

[JR13] C. S. Jutla and A. Roy. Shorter quasi-adaptive NIZK proofs for linear subspaces. In *ASIACRYPT 2013, Part I*, *LNCS* 8269, pages 1–20. Springer, Heidelberg, December 2013. (page 37.)

[KK12] S. A. Kakvi and E. Kiltz. Optimal security proofs for full domain hash, revisited. In *EUROCRYPT 2012*, *LNCS* 7237, pages 537–553. Springer, Heidelberg, April 2012. (pages 2, 25, and 30.)

[KOS10] E. Kiltz, A. O'Neill, and A. Smith. Instantiability of RSA-OAEP under chosen-plaintext attack. In *CRYPTO 2010*, *LNCS* 6223, pages 295–313. Springer, Heidelberg, August 2010. (page 5.)

[Kra00] H. Krawczyk. Simple forward-secure signatures from any signature scheme. In *ACM CCS 00*, pages 108–115. ACM Press, November 2000. (pages 34 and 35.)

[KV09] J. Katz and V. Vaikuntanathan. Signature schemes with bounded leakage resilience. In *ASIACRYPT 2009*, *LNCS* 5912, pages 703–720. Springer, Heidelberg, December 2009. (page 38.)

[KW03] J. Katz and N. Wang. Efficiency improvements for signature schemes with tight security reductions. In *ACM CCS 03*, pages 155–164. ACM Press, October 2003. (pages 1 and 29.)

[LM06] V. Lyubashevsky and D. Micciancio. Generalized compact Knapsacks are collision resistant. In *ICALP 2006, Part II*, *LNCS* 4052, pages 144–155. Springer, Heidelberg, July 2006. (page 1.)

[Mic94] S. Micali. A secure and efficient digital signature algorithm. Technical Memo MIT/LCS/TM-501b, Massachusetts Institute of Technology, Laboratory for Computer Science, April 1994. (page 1.)

[MM11] D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *CRYPTO 2011*, *LNCS* 6841, pages 465–484. Springer, Heidelberg, August 2011. (page 1.)

[MMM02] T. Malkin, D. Micciancio, and S. K. Miner. Efficient generic forward-secure signatures with an unbounded number of time periods. In *EUROCRYPT 2002*, *LNCS* 2332, pages 400–417. Springer, Heidelberg, April / May 2002. (pages 3, 22, 26, 34, and 35.)

[MR02] S. Micali and L. Reyzin. Improving the exact security of digital signature schemes. *Journal of Cryptology*, 15(1):1–18, 2002. (pages 1, 8, 20, 24, 48, 52, 53, 54, and 55.)

[MS04] A. Menezes and N. Smart. Security of signature schemes in a multi-user setting. *Designs, Codes and Cryptography*, 33(3):261–274, 2004. (pages 3 and 34.)

[OO90] K. Ohta and T. Okamoto. A modification of the Fiat-Shamir scheme. In *CRYPTO'88*, *LNCS* 403, pages 232–243. Springer, Heidelberg, August 1990. (page 1.)

[OS91] H. Ong and C.-P. Schnorr. Fast signature generation with a Fiat-Shamir-like scheme. In *EUROCRYPT'90*, *LNCS* 473, pages 432–440. Springer, Heidelberg, May 1991. (pages 1 and 29.)

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT'99*, *LNCS* 1592, pages 223–238. Springer, Heidelberg, May 1999. (pages 1 and 30.)

[PR06] C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC 2006*, *LNCS* 3876, pages 145–166. Springer, Heidelberg, March 2006. (page 1.)

[PS98] S. Patel and G. S. Sundaram. An efficient discrete log pseudo random generator. In *CRYPTO'98*, *LNCS* 1462, pages 304–317. Springer, Heidelberg, August 1998. (page 1.)

[PS00] D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000. (page 52.)

[PV05] P. Paillier and D. Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In *ASIACRYPT 2005*, *LNCS* 3788, pages 1–20. Springer, Heidelberg, December 2005. (page 1.)

[Sch90] C.-P. Schnorr. Efficient identification and signatures for smart cards (abstract) (rump session). In *EUROCRYPT'89*, *LNCS* 434, pages 688–689. Springer, Heidelberg, April 1990. (page 1.)

[Seu12] Y. Seurin. On the exact security of Schnorr-type signatures in the random oracle model. In *EUROCRYPT 2012*, *LNCS* 7237, pages 554–571. Springer, Heidelberg, April 2012. (page 1.)

[vW96] P. C. van Oorschot and M. J. Wiener. On Diffie-Hellman key agreement with short exponents. In *EUROCRYPT'96*, *LNCS* 1070, pages 332–343. Springer, Heidelberg, May 1996. (page 1.)

A Relations Between Security Notions

Micali and Reyzin introduced the $(t, q_h, q_s, \varepsilon, \delta)$-selective-security notion for signatures in [MR02] but without explaining its relation to the standard $(t, q_h, q_s, \varepsilon)$-security notion. In Section 2.5, we presented a generalization of this selective notion to the forward-security setting (W-SUF-CMA and W-EUF-CMA). In this appendix, we prove several propositions to improve our understanding of the relation between these two notions. These propositions apply to both selective forward security (W-SUF-CMA) and selective existential forward security (W-EUF-CMA). Therefore, for the sake of clarity, we focus on selective forward security.

First, we have the following straightforward proposition:

Proposition A.1 *Let $\varepsilon, \delta \in [0,1]^2$, such that $T\varepsilon\delta < 1$. A $(t, q_h, q_s, T\varepsilon\delta)$-forward-secure scheme is also $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure.*

Proof: Let $\mathcal{A}$ be an adversary which $(t, q_h, q_s, \varepsilon, \delta)$-selectively-breaks the scheme. Then, it is clear from the definitions that $\mathbf{Adv}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) \geq \varepsilon\delta$. But we also have $\mathbf{Adv}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) = \frac{1}{T}\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$. Therefore $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) \geq T\varepsilon\delta$, and $\mathcal{A}$ $(t, q_h, q_s, T\varepsilon\delta)$-breaks the scheme. ∎

Unfortunately, the converse of Proposition A.1 is not necessarily true. Let $\varepsilon, \delta, \eta \in]0,1]^3$ and $q_h \geq 1$ and suppose there exists a $(t, q_h, q_s, 0)$-secure scheme $\mathcal{FS}$.[25] By using $\mathcal{FS}$, we can construct a $(t, q_h, q_s, \varepsilon, \delta)$-selectively-secure scheme $\mathcal{FS}'$ which is not $(t, q_h, q_s, (1-\eta)\delta)$-secure. Let the scheme $\mathcal{FS}'$ be the same as $\mathcal{FS}$ except that the key generation algorithm KG includes the secret key sk_1 in the public key pk with probability $\delta(1-\eta)$. Clearly, there exists an adversary which can $(t, q_h, q_s, (1-\eta)\delta)$-break $\mathcal{FS}'$, but no adversary can $(t, q_h, q_s, \varepsilon, \delta)$-selectively-break $\mathcal{FS}'$, since only a proportion $\delta(1-\eta)$ of the keys are breakable. As a result, if a scheme is only proven to be $(t, q_h, q_s, \varepsilon, \delta)$-selectively-secure, no matter how small is ε, if $\delta' < \delta$, the scheme is not necessarily (t, q_h, q_s, δ')-secure.

We can also construct a $(t, q_h, q_s, \varepsilon, \delta)$-selectively-secure scheme $\mathcal{FS}'$ which is not $(t, q_h, q_s, (1-\eta)\varepsilon)$-secure, if there exists a scheme $\mathcal{FS}$ for which the best adversary $\mathcal{A}$ wins the game $\mathbf{Exp}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ with probability $(1-\eta)\varepsilon$ (independently of the choice of the period i, and of the key pair (pk, sk_1)).[26] As a result, if a scheme is only proven to be $(t, q_h, q_s, \varepsilon, \delta)$-selectively-secure, no matter how small is δ, the scheme is not necessarily $(t, q_h, q_s, \varepsilon')$-secure if $\varepsilon' < \varepsilon$.

After these negative results, let us now present a positive one pertaining the relation between these notions.

Proposition A.2 *Suppose there are λ different possible key pairs (pk, sk_1), and that KG chooses uniformly at random one of them.*[27] *Let $\mathcal{FS}$ be a key-evolving scheme. Let $\varepsilon' \in]0,1[$ and $\alpha = 1 + \log(\lambda T)$.*

If $\mathcal{FS}$ is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure for any $\varepsilon, \delta \in]0,1]^2$ such that $\varepsilon\delta \geq \varepsilon'/(\alpha T)$, then $\mathcal{FS}$ is $(t, q_h, q_s, \varepsilon')$-forward-secure.

Proof: Let $\mathcal{A}$ be an adversary which $(t, q_h, q_s, \varepsilon')$-breaks the scheme. Then we have $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) \geq \varepsilon'$, and as in the previous proof, we also have

$$\mathbf{Adv}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) = \mathbf{Adv}^{\text{euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})/T \geq \varepsilon'/T.$$

[25] We could use a $(t, q_h, q_s, \varepsilon')$-secure scheme with a small enough ε'. But this would make the proof more complicated.

[26] By best, we mean that, for any fixed period i and key pair (pk, sk_1), no adversary can win the game with probability greater than $(1-\eta)\varepsilon$.

[27] This is the case with most currently used schemes.

Let us consider the advantages $\mathbf{Adv}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ for each triple (pk, sk_1, i). Let us sort them in decreasing order and write them $\varepsilon'_1 \geq \cdots \geq \varepsilon'_{\lambda T}$. By hypothesis, all triples are equiprobable, therefore, we have

$$\frac{\varepsilon'}{T} \leq \mathbf{Adv}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A}) = \frac{\varepsilon'_1 + \cdots + \varepsilon'_{\lambda T}}{\lambda T}.$$

We remark that since $\varepsilon'_1 \geq \cdots \geq \varepsilon'_{\lambda T}$, if one of the j first triples is used in the game, the advantage $\mathbf{Adv}^{\text{w-euf-cma}}_{\mathcal{FS},k,T}(\mathcal{A})$ is at least ε'_j. Therefore, for any j, $\mathcal{A}$ $(t, q_h, q_s, \varepsilon'_j, j/(\lambda T))$-selectively-breaks $\mathcal{FS}$. So we just need to prove that for some j, $\varepsilon'_j j/(\lambda T) \geq \varepsilon'/(\alpha T)$. Let us suppose for all j, $\varepsilon'_j < (\lambda\varepsilon')/(\alpha j)$, by contrapositive. Then, we can sum these inequalities and we get

$$\lambda\varepsilon' \leq \sum_{j=1}^{\lambda T} \varepsilon'_j < \frac{\lambda\varepsilon'}{\alpha} \sum_{j=1}^{\lambda T} \frac{1}{j} < \lambda\varepsilon'$$

where the rightmost inequality comes from the well-known inequality for harmonic series: $\sum_{j=1}^{\lambda T} \frac{1}{j} < \alpha$. This is contradictory. So we have proven the proposition. ▮

Corollary A.3 *Under the same assumptions as in Proposition A.2, if $\mathcal{FS}$ is $(t, q_h, q_s, \varepsilon'/(\alpha T), \varepsilon'/(\alpha T))$-selectively-forward-secure, then $\mathcal{FS}$ is $(t, q_h, q_s, \varepsilon')$-forward-secure.*

Proof: We just need to remark that if $\varepsilon\delta \geq \varepsilon'/(\alpha T)$, then, since $\delta, \varepsilon \leq 1$, we have $\varepsilon, \delta \geq \varepsilon'/(\alpha T)$ and thus $\mathcal{FS}$ is also $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure. And we apply Proposition A.2. ▮

B Mathematical Tools

B.1 Results on Residues

This section presents various results on multiples (i.e., residues for an additive law) in cyclic groups and then uses them to prove results on residues of $\mathbb{Z}_N^*$, with $N \geq 3$ an odd number. In this section, $\gcd(a, b) = a \wedge b$ is the greatest common divisor (gcd) of a and b.

Multiples in cyclic groups

Let n be an integer greater or equal to 2. Let a be an element of $\mathbb{Z}_n$.

Definition.

Definition B.1 Let α be a positive integer. a is an **α-multiple** (modulo n) if and only if there exists $b \in \mathbb{Z}_n$, such that $a = \alpha b$.

Characterization of α-multiples in $\mathbb{Z}_n$. Let $\gcd(u, v) = u \wedge v$ be the greatest common divisor of two integers u and v.

Remark B.2 If β is an integer which divides n, β divides $(a \bmod n)$ if and only if it divides any $(a + \ell n)$ (where ℓ is an integer). In this case, we say that β divides a.

Theorem B.3 *Let α be a positive integer. a is an α-multiple if and only if $\gcd(\alpha, n)$ divides a.*

Proof: If a is an α-multiple, there exists $b \in \mathbb{Z}_n$ such that $a = \alpha b \bmod n$, so there exists an integer m such that α divides $a - mn$. Therefore $\gcd(\alpha, n)$ divides $a - mn$ and $\gcd(\alpha, n)$ divides a.

Suppose $d = \gcd(\alpha, n)$ divides a. There exists b such that $a = db$. Thanks to Bezout theorem, there exists two integers u and v such that $u\alpha + vn = d$. Then, in $\mathbb{Z}_n$, $a = db = db - vnb = u\alpha b$ and a is an α-multiple. ∎

Corollary B.4 *There are exactly* $\frac{n}{\gcd(\alpha,n)}$ *α-multiples modulo n. Furthermore for each α-multiples modulo n, there exists* $\gcd(\alpha, n)$ *elements $b \in \mathbb{Z}_n$, such that $a = \alpha b$.*

Proof: Thanks to the previous theorem, the α-multiples are the elements $\gcd(\alpha, n) \cdot a$, with $a \in \{0, \ldots, \frac{n}{\gcd(\alpha,n)} - 1\}$. And if a is an α-multiple, there exists b such that $a = \alpha b$, and then $a = \alpha \cdot (b + \frac{in}{\gcd(\alpha,n)})$, for each $i \in \{0, \ldots, \gcd(\alpha, n) - 1\}$. ∎

In addition, we have the following corollary, which leads to an efficient way of checking e-residuosity in $\mathbb{Z}_N^*$ in Proposition B.16:

Corollary B.5 *Let α be a positive integer. a is an α-multiple if and only if* $\frac{n}{\gcd(\alpha,n)}a = 0$ *(in $\mathbb{Z}_n$).*

Proof: Thanks to the previous theorem, it is sufficient to prove that $\frac{n}{\gcd(\alpha,n)}a = 0$ if and only if $\gcd(\alpha, n)$ divides a. If $\gcd(\alpha, n)$ divides a, clearly $\frac{n}{\gcd(\alpha,n)}a = 0$. Otherwise, let us write $a = \gcd(\alpha, n)q + r$, with $0 \leq r < \gcd(\alpha, n)$, then $\frac{n}{\gcd(\alpha,n)}a = \frac{nr}{\gcd(\alpha,n)} \neq 0$ in $\mathbb{Z}_n$. ∎

Main theorem.

Theorem B.6 *Let α, β, γ be three positive integers. Suppose γ is co-prime to n. Then, βa is a α-multiple, if and only if a is a $\gamma\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$-multiple.*

Remark B.7 Let us choose $a = \frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$. Then βa is divisible by $\alpha \wedge n$ and so is an α-multiple. But, for any divisor $\gamma \neq 1$ of n, which does not divides $\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$, a is not a γ-multiple. Hence, we can see the theorem as optimal.

Proof: If a is a $\gamma\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$-multiple, a is divisible by $\gcd(\gamma\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}, n) = \frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$ and so βa is divisible by $\alpha \wedge n$ and a is an α-multiple.

If βa is an α-multiple, βa is divisible by $\alpha \wedge n$. $\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$ is co-prime to $\frac{\beta}{\alpha\wedge\beta\wedge n}$ and divides $\frac{\beta}{\alpha\wedge\beta\wedge n}a$. So, thanks to Gauss theorem, a is divisible by $\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$. Since $\gcd(\gamma\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}, n) = \frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$, a is a $\gamma\frac{\alpha\wedge n}{\alpha\wedge\beta\wedge n}$-multiple. ∎

Residues of $\mathbb{Z}_N^*$

We can then use the previous results to prove some results on residues of $\mathbb{Z}_N^*$.

Definition B.8 Let e be a positive integer. $A \in \mathbb{Z}_N^*$ is a **e-residue** modulo N if and only if there exists $B \in \mathbb{Z}_N^*$ such that $A = B^e$.

Remark B.9 Let p be an odd prime number and k a positive integer. It is well know there exists a group isomorphism ψ_{p^k} from $\mathbb{Z}_{p^k}^*$ to $\mathbb{Z}_{p^k - p^{k-1}}$. And we can see that, for any $A \in \mathbb{Z}_{p^k}^*$, A is a e-residue modulo p^k if and only if $\psi_{p^k}(A)$ is a e-multiple (in $\mathbb{Z}_{p^k - p^{k-1}}$).

Let $N = p_1^{k_1} \ldots p_m^{k_m}$ be the prime decomposition of N.
The following lemma comes directly from the Chinese Remain Theorem:

Lemma B.10 *$A \in \mathbb{Z}_N^*$ is a e-residue modulo N if and only is it is an e-residue modulo $p_i^{k_i}$ for all i.*

And then, thanks to Theorem B.6, Remark B.9, and Lemma B.10, we have the following theorem:

Theorem B.11 *Let e, c be two positive integer, and U, Z two elements of Z_N^* such that $Z^e = U^c$ (i.e., U^c is a e-residue). Then U is a e'-residue, with e' the gcd of all $\frac{e \wedge (p_i^{k_i} - p_i^{k_i - 1})}{c \wedge e \wedge (p_i^{k_i} - p_i^{k_i - 1})} \cdot e_i$, with e_i the largest divisor of e co-prime to $p_i^{k_i} - p_i^{k_i - 1}$ (for $i \in \{1, \ldots, m\}$).*

Remark B.12 The optimality of this theorem comes from the optimality of Theorem B.6.

We recall that $\phi(N) = \prod_{i=1}^{m}(p_i^{k_i} - p_i^{k_i - 1})$ is the cardinal of $\mathbb{Z}_N^*$. Thanks to Remark B.9, Lemma B.10, and Corollary B.4, we also have the following proposition:

Proposition B.13 *The number of e-residues modulo N is:*

$$\phi(N, e) = \prod_{i=1}^{m} \frac{p^{k_i} - p^{k_i - 1}}{e \wedge (p^{k_i} - p^{k_i - 1})}$$

Furthermore, each e-residue modulo N has exactly $\phi(N)/\phi(N, e)$ roots.

When e is co-prime with $\phi(N)$, we have $\phi(N, e) = \phi(N)$, hence we get the following corollary:

Corollary B.14 *When e is co-prime with $\phi(N)$, all elements of $\mathbb{Z}_N^*$ are e-residues. Furthermore, each element of $\mathbb{Z}_N^*$ has exactly a unique root. In other words, the function f defined by $f(x) = x^e \bmod N$ is a permutation over $\mathbb{Z}_N^*$.*

When e divides $\phi(N)$, we have $\phi(N, e) = \phi(N)/e$, hence we get the following corollary:

Corollary B.15 *When e divides $\phi(N)$, the number of e-residues modulo N is $\phi(N)/e$. Furthermore, each e-residue has exactly e roots.*

And thanks to Remark B.9, Lemma B.10, and Corollary B.5, we also have the following proposition, which yields an efficient algorithm to know if an integer U is an e-residue modulo N or not (if the factorization of N is known):

Proposition B.16 *$U \in \mathbb{Z}_N^*$ is an e-residue modulo N if and only if, for all i:*

$$U^{\frac{p^{k_i} - p^{k_i - 1}}{e \wedge (p^{k_i} - p^{k_i - 1})}} = 1 \bmod p^{k_i}.$$

B.2 Some Propositions on Prime Numbers

This section shows some known results on primes numbers.

Let $\pi(x)$ be the number of primes not greater than x. The following lemma is a direct corollary of Theorem 1.10 in [Dus98, p. 36]:

Lemma B.17 *For $x \geq 599$,*

$$\frac{x}{\log x}\left(1 + \frac{1}{\log x}\right) \leq \pi(x) \leq \frac{x}{\log x}\left(1 + \frac{1.28}{\log x}\right)$$

From this lemma, we can prove the following proposition, by carefully bounding $\pi(2^{\ell_e}) - \pi(2^{\ell_e - 1})$.

Proposition B.18 *The number of primes of length ℓ_e is at least $2^{\ell_e - 1}/(\ell_e - 1)$, if $\ell_e \geq 11$.*

Let us now introduce a proposition useful to prove key indistinguishability in the GQ scheme (Section 4.1).

Proposition B.19 *Let ℓ_N, ℓ_e be two positive integers. Suppose $\ell_e < \ell_N$ and $\ell_e \geq 11$. Let N be a ℓ_N-bit integer. Let D_0 be the uniform distribution for ℓ_e-bit primes. Let D_1 be the uniform distribution for ℓ_e-bit primes, co-prime to $\phi(N)$. The statistical distance between D_0 and D_1 is at most $\frac{\ell_N+1}{2^{\ell_e-1}}$.*

Proof: Let N_0 be the number of ℓ_e-bit primes and N_1 be the number of ℓ_e-bit primes, co-prime to $\phi(N)$. We remark that a ℓ_e-bit prime is at least 2^{ℓ_e-1}, and so there are at most $(\ell_N+1)/(\ell_e-1)$ such primes which divide $\phi(N)$. Otherwise, their product would be greater than 2^{ℓ_N+1}. Since this product divides $\phi(N) < 2^{\ell_N+1}$, this is impossible. Therefore, $N_1 \geq N_0 - (\ell_N+1)/(\ell_e-1)$. Furthermore, according to Proposition B.18: $N_0 \geq 2^{\ell_e-1}/(\ell_e-1)$.

According to Lemma 2.3, the statistical distance between D_0 and D_1 is

$$1 - \frac{N_1}{N_0} = \frac{N_0 - N_1}{N_0} \leq \frac{(\ell_N+1)/(\ell_e-1)}{2^{\ell_e-1}/(\ell_e-1)} = \frac{\ell_N+1}{2^{\ell_e-1}}.$$

∎

C Proofs of Security Based on the Forking Lemma for Key-Evolving Collision-Intractable Identification Schemes

In this appendix, we give proofs of security based on the forking lemma for signatures obtained from some particular key-evolving identification schemes via the generalized Fiat-Shamir transform described in Section 3.2. This is a generalization of [MR02], which itself is based on [PS00].

C.1 Key-Evolving Collision-Intractable Identification Schemes

In this section, we extend the notion of collision-intractable identification schemes introduced in [CD95] to the key-evolving settibg. Let $\mathcal{ID}$ be a key-evolving identification scheme, as described in Section 3.1.

Informally, $\mathcal{ID}$ is collision-intractable if an adversary cannot output two valid transcripts (cmt, ch, rsp) and (cmt, ch', rsp') with $ch \neq ch'$, for a period $\tilde{\imath}$, even with access to the public key pk and the secret key $sk_{\tilde{\imath}+1}$ for period $\tilde{\imath}+1$.

More formally, let $\mathcal{A}$ be an adversary and k be a security parameter. Let $\mathbf{Exp}^{\text{col-int}}_{\mathcal{ID},k}(\mathcal{A})$ be the following experiment played between $\mathcal{A}$ and a hypothetical challenger:

$\mathbf{Exp}^{\text{col-int}}_{\mathcal{ID},k}(\mathcal{A})$
$(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k)$; $\tilde{\imath} \xleftarrow{\$} \{1, \ldots, T\}$; $sk_{\tilde{\imath}+1} = \mathsf{Update}^{\tilde{\imath}}(sk_1)$
$(cmt, ch, rsp, ch', rsp') \xleftarrow{\$} \mathcal{A}(\tilde{\imath}, pk, sk_{\tilde{\imath}+1})$
$d = \mathsf{Ver}(pk, cmt, ch, rsp, i) \wedge \mathsf{Ver}(pk, cmt, ch', rsp', \tilde{\imath}) \wedge ch \neq ch'$
return d

$\mathcal{A}$ is said to (t, ε)-breaks the collision-intractability problem if $\mathcal{A}$ runs in time at most t and its probability of success is $\Pr\left[\mathbf{Exp}^{\text{col-int}}_{\mathcal{ID},k}(\mathcal{A}) = 1\right] \geq \varepsilon$. Furthermore, $\mathcal{ID}$ is said to be (t, ε)-collision-intractable if no adversary (t, ε)-breaks the collision-intractability problem

C.2 Generalized Fiat-Shamir Transformation

Theorem C.1 *Let $\mathcal{ID} = (\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \mathcal{C}, \mathsf{Ver})$ be a key-evolving lossy identification scheme whose commitment space has min-entropy at least β (for every period i), let H be a hash function modeled as a random oracle, and let $\mathcal{FS}[\mathcal{ID}] = (\mathsf{KG}, \mathsf{Sign}, \mathsf{Ver})$ be the signature scheme obtained via the generalized Fiat-Shamir transform (Figure 3.1). If $\mathcal{ID}$ is ε_s-simulatable, complete, (t', ε')-collision-intractable, then $\mathcal{FS}[\mathcal{ID}]$ is $(t, q_h, q_s, \varepsilon, \delta)$-existentially-selectively-forward-secure in the random oracle model for:*

$$t \approx \frac{(t' - (T-1)\, t_{\mathsf{Update}}) \cdot \left(\varepsilon - q_s\, \varepsilon_s - (q_h + q_s + 1)\, q_s / 2^{\beta} - 2(q_h+1)/|\mathcal{C}|\right)}{2q_h + 3} - q_s t_{\mathsf{Sim-Sign}}$$

as long as

$$\varepsilon > q_s \varepsilon_s + \frac{(q_h + q_s + 1) q_s}{2^{\beta}} + 2\frac{q_h + 1}{|\mathcal{C}|} \qquad \text{and} \qquad \varepsilon' \leq \delta\left(1 - \frac{1}{\mathrm{e}}\right)^2$$

where $t_{\mathsf{Sim-Sign}}$ denotes the time to simulate a transcript using $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}$ and t_{Update} denotes the time to update a secret key using Update. Furthermore, if $\mathcal{ID}$ is response-unique (for normal keys),[28] $\mathcal{FS}[\mathcal{ID}]$ is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure.

Corollary C.2 *Under the same hypothesis of C.1, $\mathcal{FS}[\mathcal{ID}]$ is $(t, q_h, q_s, \varepsilon, \delta)$-selectively-forward-secure in the random oracle model for:*

$$t \approx \frac{(t' - (T-1)\, t_{\mathsf{Update}}) \cdot \varepsilon}{4q_h + 6} - q_s t_{\mathsf{Sim-Sign}}$$

as long as

$$\varepsilon \geq 2\left(q_s \varepsilon_s + \frac{(q_h + q_s + 1) q_s}{2^{\beta}} + 2\frac{q_h + 1}{|\mathcal{C}|}\right) \qquad \text{and} \qquad \varepsilon' \leq \delta\left(1 - \frac{1}{\mathrm{e}}\right)^2.$$

Proof of Corollary C.2: The condition $\varepsilon \geq 2\left(q_s\varepsilon_s + (q_h + q_s + 1)q_s/2^{\beta} + 2(q_h+1)/|\mathcal{C}|\right)$ ensures that $\varepsilon - q_s\varepsilon_s - (q_h + q_s + 1)q_s/2^{\beta} - 2(q_h+1)/|\mathcal{C}| \geq \varepsilon/2$. ∎

Proof of Theorem C.1: We use the same methods as Micali and Reyzin in [MR02]. Let us suppose there exists an adversary $\mathcal{A}$ which $(t, q_h, q_s, \varepsilon, \delta)$-breaks $\mathcal{FS}[\mathcal{ID}]$. In particular, $\mathcal{A}$ $(t, q_h, q_s, \varepsilon\delta)$-breaks $\mathcal{FS}[\mathcal{ID}]$. Let us consider the games $\mathrm{G}_0, \ldots, \mathrm{G}_7$ of Figure 3.2 and Figure 3.3, modified as for the proof of Theorem 3.3 (using **Initialize** and **Finalize** of G_6 for games $\mathrm{G}_0, \ldots, \mathrm{G}_5$), and G'_8 of Figure C.1.

As for the proof of Theorem 3.3, according to Equation (3.2), if we write $\gamma = q_s\varepsilon_s + (q_h + q_s + 1)q_s/2^{\beta}$:

$$\Pr\left[\mathrm{G}_7(\mathcal{A}) \Rightarrow 1\right] \geq \varepsilon - \gamma \text{ with probability at least } \delta \text{ over } (pk, sk_1, \tilde{\imath}).$$

In G'_8, the game outputs 0 if the signature does not correspond to the challenge ch^*. Since we have

$$\Pr\left[\mathrm{G}_7(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}_7\right] = \Pr\left[\mathrm{G}_7(\mathcal{A}) \Rightarrow 1\right] \cdot \Pr\left[\mathsf{QT}[\mathrm{fp}] = (cmt^*, M^*)\right] = \frac{1}{q_h + 1}\Pr\left[\mathrm{G}_7(\mathcal{A}) \Rightarrow 1\right],$$

and $\Pr\left[\mathrm{G}'_8(\mathcal{A}) \Rightarrow 1 \wedge \mathsf{Good}'_8\right] = \Pr\left[\mathrm{G}'_8(\mathcal{A}) \Rightarrow 1\right]$, according to Lemma 2.2, we have

$$\Pr\left[\mathrm{G}_7(\mathcal{A}) \Rightarrow 1\right] = (q_h + 1)\Pr\left[\mathrm{G}'_8(\mathcal{A}) \Rightarrow 1\right].$$

And so

$$\Pr\left[\mathrm{G}'_8(\mathcal{A}) \Rightarrow 1\right] \geq \frac{\varepsilon - \gamma}{q_h + 1} \text{ with probability at least } \delta \text{ over } (pk, sk_1, \tilde{\imath}).$$

[28] There are no lossy keys, so it does not make sense to consider response uniqueness for lossy keys, contrary to Theorem 3.1

Initialize(k, T) Game $\mathrm{G}_7, \mathrm{G}'_8$

701 $S, \mathsf{HT}, \mathsf{QT} \leftarrow []$
702 $\mathrm{hc} \leftarrow 1$; $\mathrm{b} \leftarrow T + 1$
703 $\mathrm{fp} \xleftarrow{\$} \{1, \ldots, q_h + 1\}$; $ch^* \xleftarrow{\$} \mathcal{C}$
704 $\tilde{\imath} \xleftarrow{\$} \{1, \ldots, T\}$
705 $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$
706 for $i = 1, \ldots, T-1$
707 $\quad sk_{i+1} \leftarrow \mathsf{Update}(sk_i)$
708 return (pk, T)

H(x) Game $\mathrm{G}_7, \mathrm{G}'_8$

711 if $\mathsf{HT}[x] = \perp$ then
712 $\quad \mathsf{QT}[\mathrm{hc}] \leftarrow x$
713 $\quad$ if $\mathrm{hc} \neq \mathrm{fp}$ then
714 $\quad\quad \mathsf{HT}[x] \xleftarrow{\$} \mathcal{C}$
715 $\quad$ else
716 $\quad\quad \mathsf{HT}[x] \xleftarrow{\$} ch^*$
717 $\mathrm{hc} \leftarrow \mathrm{hc} + 1$
718 return $\mathsf{HT}[x]$

Break-In(i) Game $\mathrm{G}_7, \mathrm{G}'_8$

721 if $1 \leq i \leq T$ then
722 $\quad \mathrm{b} \leftarrow \min(i, \mathrm{b})$
723 $\quad$ if $i \leq \tilde{\imath}$
724 $\quad\quad$ return $\perp$
725 $\quad$ return sk_i
726 else
727 $\quad$ return $\perp$

Sign(M, i) Game $\mathrm{G}_7, \mathrm{G}'_8$

731 $(cmt, ch, rsp) \xleftarrow{\$} \widetilde{\mathsf{Tr}}^{I\mathcal{D}}_{pk,i,k}$
732 $\mathsf{HT}[(cmt, M, i)] \leftarrow ch$
733 $\sigma \leftarrow (cmt, rsp)$
734 $S[(cmt, M, i)] \leftarrow rsp$
735 return (σ, i)

Finalize$(M^*, (\sigma^*, i^*))$ Game $\mathrm{G}_7, \boxed{\mathrm{G}'_8}$

751 $d \leftarrow \mathsf{Ver}(pk, (\sigma^*, i^*), M^*)$
752 if $i^* \geq \mathrm{b}$ or $i^* \neq \tilde{\imath}$ then
753 $\quad d \leftarrow 0$
754 $(cmt^*, rsp^*) \leftarrow \sigma^*$
755 if $\mathsf{QT}[\mathrm{fp}] \neq (cmt^*, M^*)$ then
756 $\quad \mathsf{bad} \leftarrow \mathsf{true}$
757 $\quad \boxed{d \leftarrow 0}$
758 if $S[(cmt^*, M^*, i^*)] \neq \perp$ then
759 $\quad d \leftarrow 0$
760 return d

Figure C.1: Games $\mathrm{G}_7, \mathrm{G}'_8$ for proof of Theorem C.1. G'_8 includes the boxed code at line 757 but G_7 does not.

Let $a = \frac{\varepsilon - \gamma}{q_h + 1}$. Let us now construct an adversary $\mathcal{B}$ which breaks the collision-intractability problem. In the first part, $\mathcal{B}$ runs $\mathcal{A}$ α times and simulates the oracles as in G'_8, except for **Initialize** where it uses directly its inputs $\tilde{\imath}$, pk, and $sk_{\tilde{\imath}+1}$ (instead of picking them uniformly at random). Every repetition starts completely anew, i.e., with a new random tape for $\mathcal{A}$ and new answers for the random oracle. With probability $1 - (1 - \alpha)^{\alpha^{-1}} \geq 1 - 1/\mathrm{e}$, the adversary $\mathcal{A}$ outputs at least a correct forgery accepted by **Finalize**, i.e., a forgery for period $i^* = \tilde{\imath}$ and with the challenge corresponding to the $\mathrm{fp}^{\mathrm{th}}$ query to the random oracle, fp being chosen uniformly at random at each run of $\mathcal{A}$. The adversary $\mathcal{B}$ stores the first correct output forgery $(M^*, (\sigma^*, i^*))$. Let $(cmt^*, rsp^*) = \sigma$ and $ch^* = \mathsf{H}((cmt^*, M^*))$, such that (cmt^*, ch^*, rsp^*) is a correct transcript.

Now we can run again $\mathcal{A}$ a certain number of times with the same random tape s and the same answers for the random oracle queries up to the $\mathrm{fp}^{\mathrm{th}}$ query, and then use new uniform random answers.

Let us compute the probability that $\mathcal{A}$ will again output a correct forgery. Let ξ be the random variable $(s, \mathrm{fp}, h_1, \ldots, h_{\mathrm{fp}-1})$ with s being the random tape of the adversary $\mathcal{A}$ and $h_1, \ldots, h_{\mathrm{fp}-1}$ being the answers to the $\mathrm{fp} - 1$ first queries to the random oracle, for the first run where $\mathcal{A}$ managed to output again a correct forgery. Let E be the event that the adversary $\mathcal{A}$ outputs a correct forgery if it is simulated in the environment of game G'_8. For $\lambda = (s', \mathrm{fp}', h'_1, \ldots, h'_{\mathrm{fp}-1})$, let E_λ be the event that in such simulations, the random tape of $\mathcal{A}$ is $s = s'$, the fp chosen by **Initialize** is fp', and the answers to the $\mathrm{fp} - 1$ first queries to the random oracle are $h'_1, \ldots, h'_{\mathrm{fp}}$. The events E_λ are disjoint, $\sum_\lambda \Pr[E_\lambda] = 1$, and, because of the choice of ξ, we also have $\Pr[\xi = \lambda] = \Pr[E_\lambda \mid E]$. In addition $\Pr[E] \geq \alpha$.

We can then apply the following lemma stated and proven in [MR02] (Lemma 3).

Lemma C.3 ([MR02]) *Let E be an event with probability α. Let Λ a finite set and let $(E_\lambda)_{\lambda\in\Lambda}$ be disjoint events such that $\sum_\lambda \Pr[E_\lambda] = 1$. Let ξ be a Λ-valued random variable with the following distribution $\Pr[\xi = \lambda] = \Pr[E_\lambda \mid E]$. Then*

$$\Pr_\xi\left[\Pr[E \mid E_\xi] \geq \frac{\alpha}{2}\right] \geq \frac{1}{2}.$$

Therefore, we have $\Pr_\xi\left[\Pr[E \mid E_\xi] \geq \frac{\alpha}{2}\right] \geq \frac{1}{2}$. which means that with probability $1/2$, the probability α' that the adversary $\mathcal{A}$ will do a forgery under condition ξ is at least $\alpha/2$. Assume ξ is such that $\alpha' \geq \alpha/2$. Then the probability that $\mathcal{A}$ will output a forgery corresponding to a "good" transcript (cmt^*, ch'^*, rsp'^*) with $ch'^* \neq ch^*$ is at least $\alpha/2 - 1/|\mathcal{C}|$.

So, in the second part, $\mathcal{B}$ runs $\mathcal{A}$ $(\alpha/2 - 1/|\mathcal{C}|)^{-1}$ times under the condition ξ. The probability that $\mathcal{A}$ will output a forgery corresponding to a "good" transcript is

$$\left(1 - (1 - (\alpha/2 - 1/|\mathcal{C}|))^{(\alpha/2-1/|\mathcal{C}|)^{-1}}\right) \geq 1 - 1/\mathrm{e}.$$

Therefore, with probability $(1 - 1/\mathrm{e})^2/2$, $\mathcal{B}$ gets two transcripts (cmt^*, ch^*, rsp^*) and (cmt^*, ch'^*, rsp'^*) with $ch^* \neq ch'^*$. This corresponds to the expected output for the game of collision-intractability.

We can now slightly improve the running time of $\mathcal{B}$. Instead of simulating the environment of G'_8, let $\mathcal{B}$ simulates the environment of G_7 when it runs $\mathcal{A}$, in the first part. The only difference is that $\mathcal{B}$ accepts any forgery (in the first part) instead of accepting only forgeries for the $\mathrm{fp}^{\mathrm{th}}$ query to the random oracle, where fp is chosen uniformly at random. Therefore, $\mathcal{B}$ just needs to run $\mathcal{A}$ $\frac{1}{\varepsilon-\gamma}$ times instead of $\frac{q_h+1}{\varepsilon-\gamma}$, to have a forgery with probability at least $1 - 1/\mathrm{e}$. As explained in [MR02], the probability distribution of ξ is still the same and so it does not change the rest of the proof.

Let us now analyze the running time of $\mathcal{B}$. The first part takes about $\frac{1}{\varepsilon-\gamma}(t + q_s t_{\mathsf{Sim-Sign}})$ and the second part takes about $\frac{1}{\alpha/2-1/|\mathcal{C}|}(t + q_s t_{\mathsf{Sim-Sign}})$. Therefore $\mathcal{B}$ (t', ε')-breaks the collision intractability with

$$\begin{aligned}
t' &\leq \left(\frac{1}{\varepsilon-\gamma} + \frac{1}{(\varepsilon-\gamma)/(2(q_h+1)) - 1/|\mathcal{C}|}\right)(t + q_s t_{\mathsf{Sim-Sign}}) + (T-1)\, t_{\mathsf{Update}} \\
&\leq \left(\frac{1}{\varepsilon-\gamma-2(q_h+1)/|\mathcal{C}|} + \frac{2(q_h+1)}{\varepsilon-\gamma-2(q_h+1)/|\mathcal{C}|}\right)(t + q_s t_{\mathsf{Sim-Sign}}) + (T-1)\, t_{\mathsf{Update}} \\
&\leq \frac{(2q_h+3)(t + q_s t_{\mathsf{Sim-Sign}})}{\varepsilon-\gamma-2(q_h+1)/|\mathcal{C}|} + (T-1)\, t_{\mathsf{Update}}
\end{aligned}$$

and

$$\varepsilon' = \delta\left(1 - \frac{1}{\mathrm{e}}\right)^2 \quad \text{as long as} \quad \varepsilon \geq \gamma + 2(q_h+1)/|\mathcal{C}|.$$

We remark that this is exactly the bound of [MR02] (if $T_2 = 0$ in their paper, and $t_{\mathsf{Update}} = 0$), which is not surprising since selective existential forward security is very close to the classical existential unforgeability. ∎

C.3 Security of the Itkis-Reyzin Scheme

In this section, we apply the previous generic results to present another security analysis of the original Itkis-Reyzin scheme in [IR01], based on the forking lemma.

According to Appendix C, and more precisely to Corollary C.2, we just need to prove that the underlying key-evolving identification scheme is collision-intractable. Informally, this means that it is hard for an adversary to find two correct transcripts (cmt, ch, rsp) and (cmt, ch', rsp') for a period $\tilde{\imath}$ such that $ch \neq ch'$, given the public key pk, the period $\tilde{\imath}$, and the secret key $sk_{\tilde{\imath}+1}$ for period $\tilde{\imath}+1$. For the

IR scheme, it is straightforward to see that the identification scheme is (t', ε')-collision intractable if the strong RSA problem is (t', ε')-hard. It is also response-unique exactly for the same reason as our scheme in Section 6.3. Therefore, Theorem 5.1 follows from Corollary C.2.