Tighter Reductions for Forward-Secure Signature Schemes

Michel Abdalla, Fabrice Ben Hamouda, and David Pointcheval

Departement d'Informatique, École normale supérieure, Paris, France.
{Michel.Abdalla,Fabrice.Ben.Hamouda,David.Pointcheval}@ens.fr
http://www.di.ens.fr/users/{mabdalla,fbenhamo,pointche}

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. Finally, we 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. All of our results hold in the random oracle model.

1 Introduction

A common paradigm for constructing signature schemes is to apply the Fiat-Shamir transform [9] 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 [1], 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 [10,27,31] and obtain a signature scheme with a tighter proof of security. For instance, in [22], 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 [20], 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 [8,9,12,24,25]. 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 with respect to the commitment when computing the response, such as in [30].

In 2003, Katz and Wang [17] 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.* [3], who presented three new signature schemes based on the hardness of the short exponent discrete log problem [28,32], on the worst-case hardness of the shortest vector problem in ideal lattices [18,29], and on the hardness of the Subset Sum problem [14,23]. Additionally, they also formalized the intuition in [17] by introducing the notion of lossy identification schemes and showing that any such schemes 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 [22] could be applied in this setting (resulting in a slightly different scheme), our first contribution is to show that these signature schemes admit tight security reductions to certain decisional complexity assumptions such as the quadratic-residuosity, the high-residuosity [26], and the ϕ-hiding [6] assumptions. We do so by showing that the underlying identification schemes used in these schemes are a particular case of a lossy identification scheme [3]. As shown in Section 4.1 in the case of the Guillou-Quisquater signature scheme [12], 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 [6] assumption.

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 factor of 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 [15] (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, namely 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 2^t-strong-residuosity.

Organization. After recalling some definitions in Section 2, we introduce the notion of key-evolving lossy identification scheme and show how to transform such a scheme into a forward-secure signature scheme in Section 3. Then, in Section 4, we apply our security proof methodology to two cases: the Guillou-Quisquater scheme and its extension to the forward-secure case (i.e., our variant of the Itkis-Reyzin scheme). In Section 5, we compare this second scheme with the original Itkis-Reyzin scheme and the MMM scheme by Malkin, Micciancio and Miner [19]. Finally, we introduce our generic lossy key-evolving identification scheme and show various instantiations of it in Section 6.

2 Definitions

2.1 Notation and Conventions

Let $\mathbb{N}$ denote the set of natural numbers. 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 $\perp$. 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 $x \overset{\$}{\leftarrow} S$ 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 $y \overset{\$}{\leftarrow} \mathcal{A}(x)$ denotes that the output of an execution of $\mathcal{A}(x)$ with fresh coins 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. All our schemes are in the random oracle model [5].

2.2 Complexity Assumptions

The security of the signature schemes being analyzed in this paper will be based on decisional assumptions over composite-order groups: the e-residuosity assumption, the ϕ-hiding assumption and a new assumption called the strong-2^t-residuosity. We also need to recall the strong-RSA assumption to be able to compare our scheme with the Itkis-Reyzin scheme [15].

Let N be the product of distinct large primes p_1 and p_2. We call such N an RSA modulus. Informally, the **e-residuosity** assumption states that the problem of deciding whether a given element y in $\mathbb{Z}_N^*$ is an e-residue or not is intractable without knowing the factorization of N. Remember that an element $y \in \mathbb{Z}_N^*$ is said to be an e-residue if there exists an element $x \in \mathbb{Z}_N^*$ such that $y = x^e \bmod N$. If $e = 2$, this assumption is called the **quadratic-residuosity** assumption. Furthermore, if we extend it to $N = e^2$, with e an RSA modulus, this is called the **high-residuosity** assumption [26]. Likewise, the **ϕ-hiding** assumption, introduced by Cachin, Micali, and Stadler in [6], states that it is hard for an adversary to tell whether a prime number e divides the order of the group $\mathbb{Z}_N^*$ or not. Next, we introduce the **strong-2^t-residuosity** assumption that states that it is hard for an adversary to decide whether a given element y in $\mathbb{Z}_N^*$ is a 2^t-residue or is even not a 2-residue, when 2^t divides $p_1 - 1$ and $p_2 - 1$. Finally, the **strong-RSA** assumption states that, given an element $y \in \mathbb{Z}_N^*$, it is hard for an adversary to find an integer $e \geq 2$ and an element $x \in \mathbb{Z}_N^*$ such that $y = x^e \bmod N$.

For each of these assumptions, the underlying problem is said to be (t, ε)-hard, if no adversary running in time at most t is able to solve the problem with probability at least ε. Formal descriptions of the assumptions can be found in the full version [2].

2.3 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 [4]. 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 $\langle \sigma, i \rangle \xleftarrow{\$} \mathsf{Sign}(sk_i, M)$, the user in possession of the secret key sk_i associated with time period $i \leq T$ can generate a signature $\langle \sigma, i \rangle$ for a message $M \in \mathcal{M}$ for period i. Finally, via $d \leftarrow \mathsf{Ver}(pk, \langle \sigma, i \rangle, M)$, 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.

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 breakin period) and to signed messages of his 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) [11] to key-evolving signature scheme and a slightly stronger variant of the definition in [4]. In particular, we do not restrict the adversary to only perform signing queries with respect to the current time period.

In the remainder of the paper, we also use a stronger notion: **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, a key-evolving signature scheme is $(t, q_h, q_s, \varepsilon)$-(existentially)-forward-secure if no adversary 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 can break the (existential) forward security with probability at least ε. All the formal security notions and the comparison with [4], together with other security notions (used for detailed comparisons), can be found in the full version [2].

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 [3], which can be transformed to tightly secure signature scheme, and key-evolving identification schemes [4], 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.

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 $(\mathit{cmt}, \mathit{ch}, \mathit{rsp})$), of the public key, and of the index of the current time period.

Informally, a lossy key-evolving identification scheme has $T+1$ kinds 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 a 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 $(\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \ell_c, \mathsf{Ver})$ 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 a 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.

- ℓ_c is a polynomial; $\ell_c(k)$ (often simply denoted ℓ_c) is the length of the challenge sent by the verifier.
- 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 $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$
    $(cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)$ ; $ch \xleftarrow{\$} \{0,1\}^{\ell_c}$ ; $rsp \xleftarrow{\$} \mathsf{Prove}(sk_i, cmt, ch, st)$
    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, $\mathcal{ID}$ is said to be ε-simulatable if there exists a probabilistic polynomial time algorithm $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,i,k}$ with no access to any secret key, which can generate transcripts $\{(cmt, ch, rsp)\}$ whose distribution is statistically indistinguishable from the transcripts output by $\mathsf{Tr}^{\mathcal{ID}}_{pk,sk_i,k}$, where ε is an upper-bound for the statistical distance. When $\varepsilon = 0$, then $\mathcal{ID}$ is said to be simulatable.

(3) **Indistinguishability of keys.** Consider the two following experiments $\mathbf{Exp}^{\text{ind-keys-real}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$ and $\mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$ $(i \in \{1,\dots,T\})$:

```
$\mathbf{Exp}^{\text{ind-keys-real}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$
    $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$
    $sk_{i+1} \xleftarrow{\$} \mathsf{Update}^i(sk_1)$
    return $\mathsf{D}_i(pk, sk_{i+1})$

$\mathbf{Exp}^{\text{ind-keys-lossy}}_{\mathcal{ID},k,i}(\mathsf{D}_i)$
    $(pk, sk_{i+1}) \xleftarrow{\$} \mathsf{LKG}(1^k, 1^T, i)$
    return $\mathsf{D}_i(pk, sk_{i+1})$
```

D is said to (t, ε)-solve the key-indistinguishability problem for period i if it runs in time t and

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

Furthermore, we say that $\mathcal{ID}$ is (t, ε)-key-indistinguishable, if, for any i, no algorithm (t, ε)-solves the key-indistinguishability problem for period i.

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

```
KG(1^k, 1^T)
    $(pk, sk_1) \xleftarrow{\$} \mathsf{KG}(1^k, 1^T)$
    return $(pk, sk_1)$

Sign(sk_i, M)
    $(cmt, st) \xleftarrow{\$} \mathsf{Prove}(sk_i)$
    $ch \leftarrow \mathsf{H}(\langle cmt, M, i\rangle)$
    $rsp \xleftarrow{\$} \mathsf{Prove}(sk_i, cmt, ch, st)$
    $\sigma \leftarrow (cmt, rsp)$
    return $\langle \sigma, i\rangle$

Update(sk_i)
    $sk \leftarrow \mathsf{Update}(sk_i)$
    return $sk$

Ver(pk, ⟨σ, i⟩, M)
    $(cmt, rsp) \leftarrow \sigma$
    $ch \leftarrow \mathsf{H}(\langle cmt, M, i\rangle)$
    $d \leftarrow \mathsf{Ver}(pk, cmt, ch, rsp, i)$
    return $d$
```

Fig. 1. Generalized Fiat-Shamir transform for forward-secure signature

```
$\mathbf{Exp}^{\text{los-imp-pa}}_{\mathcal{ID},k,i}(\mathsf{I}_i)$
    $(pk, sk_{i+1}) \xleftarrow{\$} \mathsf{LKG}(1^k, 1^T, i)$ ; $(cmt, st) \xleftarrow{\$} \mathsf{I}_i(pk, sk_{i+1})$
    $ch \xleftarrow{\$} \{0,1\}^{\ell_c}$ ; $rsp \xleftarrow{\$} \mathsf{I}_i(ch, st)$
    return $\mathsf{Ver}(pk, cmt, ch, rsp, i)$
```

I_i is said to ε-solve the impersonation problem with respect to i-lossy public keys if $\Pr\left[\mathbf{Exp}^{\text{los-imp-pa}}_{\mathcal{ID},k,i}(\mathsf{I}_i) = 1\right] \geq \varepsilon$. Furthermore, $\mathcal{ID}$ 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.

We remark that, for $T = 1$, a key-evolving lossy identification scheme becomes a standard lossy identification scheme[1], described in [3].

Finally, we say that $\mathcal{ID}$ is **response-unique** if for all normal public keys pk or for all lossy keys pk, for all periods $i \in \{1, \ldots, T\}$, for all messages M, for all bit strings cmt[2], 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 [9], depicted in Fig. 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. The following theorem is a generalization of (a special case of) Theorem 1 in [3], where we assume perfect completeness.

[1] Contrary to the definition of lossiness given in [3], the impersonator I_1 does not have access to an oracle $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,1,k}$ in $\mathbf{Exp}^{\text{los-imp-pa}}_{\mathcal{ID},k,1}(\mathsf{I}_1)$. 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.

[2] Not necessarily a correctly generated commitment, but any bit string.

Theorem 1. *Let $\mathcal{ID} = (\mathsf{KG}, \mathsf{LKG}, \mathsf{Update}, \mathsf{Prove}, \ell_c, \mathsf{Ver})$ be a key-evolving lossy identification scheme whose commitment space has min-entropy at least β (for every period i), let H be a 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_k)-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:*

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

where $t_{\mathsf{Sim-Sign}}$ denotes the average time of a query to the simulated transcript function $\widetilde{\mathsf{Tr}}^{\mathcal{ID}}_{pk,i,k}$ and t_{Update} denotes the average time of a query to Update. *Furthermore, if $\mathcal{ID}$ is response-unique, $\mathcal{FS}[\mathcal{ID}]$ is also $(t, q_h, q_s, \varepsilon)$-forward-secure.*

Actually, if we choose $T = 1$ in the previous theorem, we get a slightly improved special case of Theorem 1 in [3], since the forward security for $T = 1$ is exactly the strong unforgeability for a signature scheme. The proof of this theorem can be found in the full version [2] and is very similar to the proof in [3], except that we need to guess the period i^* of the signature output by the adversary, in order to choose the correct lossy key. That is why we lose a factor T in the reduction.

Remark 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.

4 Tighter Security Reductions for Guillou-Quisquater-like Schemes

In this section, we prove tighter security reductions for the Guillou-Quisquater scheme (GQ, [12]) and for a slight variant of the Itkis-Reyzin scheme (IR, [15]), 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. Detailed proofs for these schemes are available in the full version [2].

4.1 Guillou-Quisquater Scheme

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

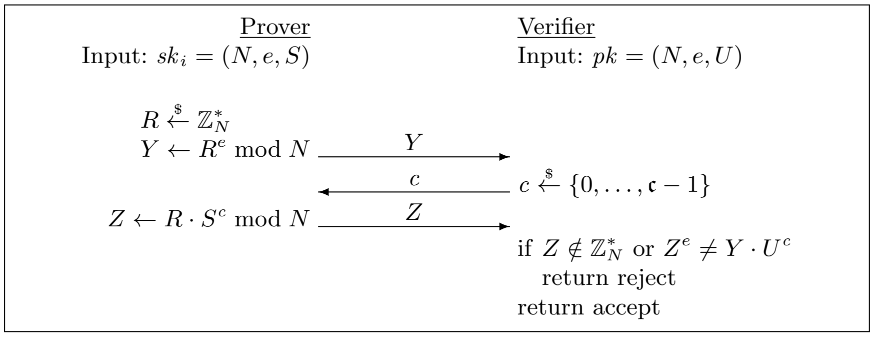
Prover
Input: $sk_i = (N, e, S)$
Verifier
Input: $pk = (N, e, U)$
$R \overset{\$}{\leftarrow} \mathbb{Z}_N^*$
$Y \leftarrow R^e \bmod N$
Y
c
$c \overset{\$}{\leftarrow} \{0, \ldots, \mathfrak{c} - 1\}$
$Z \leftarrow R \cdot S^c \bmod N$
Z
if $Z \notin \mathbb{Z}_N^*$ or $Z^e \neq Y \cdot U^c$
return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

Fig. 2. Description of the GQ identification scheme ($U = S^e \bmod N$).

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, coprime with $\phi(N) = (p_1 - 1)(p_2 - 1)$, chosen uniformly at random. Let S be an element chosen uniformly at random in $\mathbb{Z}_N^*$ and let $U = S^e \bmod N$. Let $\mathfrak{c} = 2^{\ell_e}$. The public key is $pk = (N, e, U)$ and the secret key is $sk = (N, e, S)$.

The goal of the identification scheme is to prove U is a e-residue. The identification scheme is depicted in Fig. 2 and works as follows. 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[3].

Security. The previous proofs of the GQ schemes looses a factor q_h in the reduction. In this paragraph, we prove the previously described identification scheme $\mathcal{ID}$ 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.

The algorithm LKG chooses e and $N = p_1 p_2$ such that e divides $p_1 - 1$, instead of being coprime with $\phi(N)$, and chooses U uniformly at random among the non-e-residue modulo N. In the full version [2], we 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 $k \in \{1, 2\}$, e does not divide $p_k - 1$ or $U^{(p_k - 1)/e} = 1 \bmod p_k$.

The proof that $\mathcal{ID}$ 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 $Z^e = (R \cdot S^c)^e = R^e \cdot (S^e)^c = Y \cdot U^c$.

The **simulatability** of $\mathcal{ID}$ 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

[3] The test $Z \in \mathbb{Z}_N^*$ can be replaced by the less expensive test $Z \bmod N \neq 0$, as explained in the full version [2].

done by choosing Z uniformly at random in $\mathbb{Z}_N^*$ and c uniformly at random in $\{0,\dots,\mathfrak{c}-1\}$, and setting $Y = Z^e/U^c$.

Let us prove the **key indistinguishability**. The distribution of normal public keys is 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 $1-1/e$, so this distribution is statistically close to the distribution of lossy keys. Therefore, $\mathcal{ID}$ is key indistinguishable.

To show that $\mathcal{ID}$ 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 $c \in \{0,\dots,\mathfrak{c}-1\}$ for which there exists a response Z which 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 $\{0,\dots,\mathfrak{c}-1\}$ 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^{ℓ_e}, it is coprime with e (since e is a prime and $e \geq 2^{\ell_e}$). Therefore, according to Bezout theorem, there exists two integers u, v such that: $ue + v(c_1 - c_2) = 1$. So:

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

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 $\mathcal{ID}$ is $1/\mathfrak{c}$-lossy.

Comparison with the swap method. Applying the swap method [22] to the GQ identification scheme can also provide a signature with a tight reduction, to the RSA problem. However, 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 exponentiation 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. And our signing algorithm will be $\ell_N/(2\ell_e)$ faster, for the same parameters and the same security level, if we consider the ϕ-hiding problem is as hard as the RSA problem. 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.

A slight variant of the scheme. We can also chooses e uniformly at random among the ℓ_e-bit primes (without forcing that e is coprime with $\phi(N)$ in KG), because, with high probability, such a prime number will be coprime with $\phi(N)$.

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 $e_1,\dots,e_T$ be T distinct ℓ_e-bit primes chosen uniformly at random. Let $f_i = e_{i+1}\dots e_T$, $f_T = 1$ and $E = e_1 \dots e_T$. 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, 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 the public key and in the secret key. Therefore, the keys are linear in T, the number of periods. It is possible to have constant-size key, either by using fixed exponents, 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 coprime with $\phi(N)$, 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.

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 [15] and the MMM scheme [19].

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 to the ϕ-hiding assumption, we can use a second random oracle H' which outputs prime numbers of length ℓ_e, and set $e_i = \mathsf{H}'(i)$.

An implementation of a random oracle for prime numbers using a classical random oracle is presented in the full version [2]. The construction is close to the construction of a PRF mapping to prime numbers in [13]. The idea is to hash the input value concatenated to a counter and to increment the counter until we get a prime number. One can prove that it behaves like a random oracle uniform over all primes, and that we can program it efficiently (property which is needed for the security reductions).

We finally remark that, we can always store e_i in the secret key for period 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.

Table 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

5.2 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 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 (please refer to the full version [2] for more details). In both cases, ε_p denotes the maximum error probability of the probabilistic primality test used in the random oracle for primes 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. In the sequel, all the parameters are fixed except the length ℓ_N of the modulus.

5.3 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[4] 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 breakin 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. However, we remark in the full version [2] that the other optimizations of the original IR scheme can also be applied to our scheme.

Let us now compare the two schemes with the same security parameters (k, ℓ_e, ℓ_N), before analyzing the exact security. We first remark that for the same security parameters, 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

[4] 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 2. Time of verification algorithm (using parameters of Table 1)

		exponentiation		prime generation		verification orig.[a]		verification new[b]	
k	ℓ_N	mul.[c]	ms[d]	mul.[c]	ms[d]	mul.[c]	ms[d]	mul.[c]	ms[d]
k	ℓ_N	$\frac{3}{2}\,\ell_e\,{\ell_N}^2$	n/a	$(\frac{3}{2}\,kp + 2\,\ell_e)\,{\ell_e}^3$	n/a	$3\,\ell_e\,{\ell_N}^2$	n/a	$3\,\ell_e\,{\ell_N}^2 + (\frac{3}{2}\,kp + 2\,\ell_e)\,{\ell_e}^3$	n/a
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
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**
80	6848	$8.65 \cdot 10^9$	3.09	$0.68 \cdot 10^9$	0.26	$17.3 \cdot 10^9$	**6.18**	$1.26 \cdot 10^9$	6.44
128	3248	$2.71 \cdot 10^9$	1.19	$2.67 \cdot 10^9$	0.82	$5.42 \cdot 10^9$	2.38	$8.09 \cdot 10^9$	3.10

[a] verification time of the original scheme (also equal to the signature time for both schemes), estimated using the time of the two exponentiations.
[b] verification time of our scheme, estimated using the time of the two exponentiations and of the prime generation.
[c] approximate theoretical complexity (see the full version [2]).
[d] time on an Intel Core i5 750 (2.67 GHz), using GMP version 5.0.4 (`http://gmplib.org`, a pseudo-random number generator is used as a random oracle.

exponentiations (modulo N with a ℓ_e-bit exponent) for the original scheme and two exponentiations 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 [16], 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). As shown in the full version [2], with our choice of parameters, if we want $k = 80$ bits of security, we need to choose a modulo length ℓ_N such that the factorization is $k + \log_2(T) = 100$-bit hard (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 [7]. In this case, according to Table 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 [19] 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 **Breakin**). 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 approximatively the same proven security. And the comparison of the MMM scheme with our scheme is roughly the same as the comparison in [19] 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 private 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 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 smartcard), our scheme is better than the MMM scheme. And, even more generally, if key updates are not performed often and if T can be bounded by a reasonable constant (for example, if keys are updated each day and are expected to last 3 years, $T = 2^{10}$, and key update time is not really a problem), our scheme is also better than the MMM scheme.

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

Let ℓ be a security parameter, let N be a product of large primes, 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 a set of elements in ${\mathbb{Z}_N^*}^\ell$ and let $U_1, \ldots, U_\ell \in {\mathbb{Z}_N^*}^\ell$ be the set of elements containing the corresponding E-powers. That is, for each $j \in \{1, \ldots, \ell\}$, $U_j = S_j^E \bmod N$. The public key is $pk = (N, e_1, \ldots, e_T, U_1, \ldots, U_\ell)$ (as for our variant of the IR scheme, we can use a random oracle to avoid storing the exponents in the keys, as explained in Section 5.1). 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 $S_{j,i} = S_j^{E/e_i}$

<u>Prover</u>		<u>Verifier</u>
$sk_i = (N, e_1, \dots, e_T, S_{1,i}, \dots, S_{\ell,i}, \dots)$		$pk = (N, e_1, \dots, e_T, U_1, \dots, U_\ell)$
for $j = 1, \dots, \ell$		
$\quad R_j \overset{\$}{\leftarrow} \mathbb{Z}_N^*$		
$\quad Y_j \leftarrow R_j^{e_i} \bmod N$	$\xrightarrow{Y_1, \dots, Y_\ell}$	
	$\xleftarrow{c_1, \dots, c_\ell}$	$c_1, \dots, c_\ell \overset{\$}{\leftarrow} \{0, \dots, \mathfrak{c} - 1\}^\ell$
for $j = 1, \dots, \ell$		
$\quad Z_j \leftarrow R_j \cdot S_{j,i}^{c_j} \bmod N$	$\xrightarrow{Z_1, \dots, Z_\ell}$	
		for $j = 1, \dots, \ell$
		$\quad$ if $Z_j \notin \mathbb{Z}_N^*$ or $Z_j^{e_i} \neq Y_j \cdot U_j^{c_j}$
		$\qquad$ return reject
		return accept

Fig. 3. Description of the generic identification scheme $\mathcal{ID}$ for proving that the elements $U_1, \dots, U_\ell$ in pk are all e_i-residues (for each $j \in \{1, \dots, \ell\}$, $U_j = S_{j,i}^{e} \bmod N$).

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 $sk_i = (i, N, e_i, \dots, e_T, S_{1,i}, \dots, S_{\ell,i}, S'_{1,i}, \dots, S'_{\ell,i})$. We remark that it is possible to compute sk_{i+1} from sk_i by computing: $S_{j,i+1} = S_{j,i}'^{f_i/e_{i+1}} \bmod N$ and $S'_{j,i+1} = S_{j,i}'^{f_i/f_{i+1}} \bmod N$.

The identification scheme is depicted in Fig. 3 and is a straightforward extension of the one of our variant of the IR scheme in Section 4.2. For period i, its goal is to prove that the elements $U_1, \dots, U_\ell$ are all e_i-residues, and 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 $j \in \{1, \dots, \ell\}$. It then sends $Y_1, \dots, Y_\ell$ to the verifier, which in turn chooses $c_1, \dots, c_\ell \in \{0, \dots, \mathfrak{c}-1\}^\ell$ and returns it to the prover. Upon receiving $c_1, \dots, c_\ell$, the prover computes $Z_j \leftarrow R_j \cdot S_{j,i}^{c_j} \bmod N$ for $j \in \{1, \dots, \ell\}$ 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 $j \in \{1, \dots, \ell\}$ and accepts only if this is the case. The corresponding factoring-based forward-secure signature scheme is depicted in Fig. 4.

In the full version [2], we prove that the previous scheme is existentially forward-secure, under the following condition:

Condition 3. *There exists a normal key generation algorithm* KG *and a lossy key generation algorithm* LKG *which takes as input the security parameter and the period i and outputs a pair (pk, sk'_{i+1}) such that, for every $i \in \{1, \dots, T\}$:*

- *(pk, sk'_{i+1}) is indistinguishable from a pair (pk, sk_{i+1}) generated by* KG *and i calls to* Update *(to get sk_{i+1} from sk_1);*
- *for all $c \in \{0, \dots, \mathfrak{c} - 1\}$, none of $U_1, \dots, U_\ell$ is a $e'(e, c, N)$-residue, where $e'(e, c, N)$ is:*

$$e'(e, c, N) = \gcd_{i \in \{1, \dots, m\}} \frac{e \wedge (p_i^{k_i} - p_i^{k_i - 1})}{c \wedge e \wedge (p_i^{k_i} - p_i^{k_i - 1})} e'_i,$$

```
$\mathsf{KG}(1^k, 1^T)$
Generate $N, e_1, \ldots, e_T$
$E \leftarrow \mathrm{lcm}(e_1, \ldots, e_T)$
for $i = 1, \ldots, T$
    $f_i \leftarrow \mathrm{lcm}(e_{i+1}, \ldots, e_T)$
for $j = 1, \ldots, \ell$
    $S_j \xleftarrow{\$} \mathbb{Z}_N^*$
    $S_{j,1} \leftarrow S_j^{E/e_1} \bmod N$
    $S'_{j,1} \leftarrow S_j^{E/f_1} \bmod N$
    $U_j \leftarrow S_j^{E} \bmod N$
$pk \leftarrow (N, e_1, \ldots, e_T,$
        $U_1, \ldots, U_\ell)$
$sk_1 \leftarrow (1, N, e_1, \ldots, e_T,$
        $S_{1,1}, \ldots, S_{\ell,1},$
        $S'_{1,1}, \ldots, S'_{\ell,1})$
return $(pk, sk_1)$
```

```
$\mathsf{Update}(sk, M)$
$(i, N, e_1, \ldots, e_T,$
 $S_{1,i}, \ldots, S_{\ell,i},$
 $S'_{1,i}, \ldots, S'_{\ell,i}) \leftarrow sk$
if $i = T$ then
    return $\perp$
$f_i \leftarrow \mathrm{lcm}(e_{i+1}, \ldots, e_T)$
$f_{i+1} \leftarrow \mathrm{lcm}(e_{i+2}, \ldots, e_T)$
for $j = 1, \ldots, \ell$
    $S_{j,i+1} \leftarrow S'^{f_i/e_{i+1}}_{j,i}$
    $S'_{j,i+1} \leftarrow S'^{f_i/f_{i+1}}_{j,i}$
$sk_{i+1} \leftarrow (i+1, N, e_{i+1}, \ldots, e_T,$
        $S_{1,i+1}, \ldots, S_{\ell,i+1},$
        $S'_{1,i+1}, \ldots, S'_{\ell,i+1})$
return $sk_{i+1}$
```

```
$\mathsf{Ver}(pk, \langle\sigma, i\rangle, M)$
$(N, e_1, \ldots, e_T,$
 $U1, \ldots, U_\ell) \leftarrow pk$
$((Y_1, \ldots, Y_\ell), (Z_1, \ldots, Z_\ell)) \leftarrow \sigma$
$(c_1, \ldots, c_\ell) \leftarrow \mathsf{H}(\langle(Y_1, \ldots, Y_\ell), M, i\rangle)$
for $j = 1, \ldots, \ell$
    if $Z_j \notin \mathbb{Z}_N^*$ or $Z_j^{e_i} \neq Y_j \cdot U_j^{c_j}$ then
        return reject
return accept
```

```
$\mathsf{Sign}(sk, M)$
$(i, N, e_i, \ldots, e_T,$
 $S_{1,i}, \ldots, S_{\ell,i},$
 $S'_{1,i}, \ldots, S'_{\ell,i}) \leftarrow sk$
for $j = 1, \ldots, \ell$
    $R_j \xleftarrow{\$} \mathbb{Z}_N^*$
    $Y_j \leftarrow R_j^{e_i} \bmod N$
$(c_1, \ldots, c_\ell) \leftarrow \mathsf{H}(\langle(Y_1, \ldots, Y_\ell), M, i\rangle)$
for $j = 1, \ldots, \ell$
    $Z_j \leftarrow R_j \cdot S_j^{c_j} \bmod N$
$\sigma \leftarrow ((Y_1, \ldots, Y_\ell), (Z_1, \ldots, Z_\ell))$
return $\langle\sigma, i\rangle$
```

Fig. 4. Factoring-based forward-secure signature scheme

with $N = p_1^{k_1} \ldots p_m^{k_m}$ the prime decomposition of N and e'_i the greatest divisor of e coprime with $p_i^{k_i} - p_i^{k_i - 1}$, and where $a \wedge b$ is the greatest common divisor (gcd) of a and b.

The second part of the condition ensures that the scheme is $1/\mathfrak{c}^\ell$-lossy.

6.2 Some Instantiations

In addition to the GQ scheme and our variant of the IR scheme, there are other possible instantiations of our generic scheme.

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 [17]. This scheme is existentially unforgeable 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.

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 [25]. If $N = p_1 p_2$ is an RSA modulus such that 2^t divides $p_1 - 1$ and $p_2 - 1$, this scheme is existentially unforgeable based on the hardness of the strong-2^t-residuosity problem as long as t is large enough to make the term $q_h/2^t$ 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 [26]. This scheme is existentially unforgeable based on the hardness of the high-residuosity problem of [26].

2^t-Root Forward-Secure Signature Scheme. The case in which $e_i = 2^{t(T-i+1)}$ with t a positive integer and $\mathfrak{c} = 2^i$ 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 $S'_{1,i}$. If $N = p_1 p_2$ is an RSA modulus such that 2^{tT} divides $p_1 - 1$ and $p_2 - 1$, this scheme is existentially forward-secure based on the hardness of a variant of the strong-2^{tT}-assumption, as long as the exponents t and ℓ are large enough to make the term $q_h/2^{t\ell}$ negligible. 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.

Proof details for the above instantiations can be found in the full version [2].

Acknowledgments

We would like to thank Mihir Bellare and Eike Kiltz for their helpful comments on a preliminary version of this paper and the anonymous referees of PKC 2013 for their valuable input.

This work was supported in part by the French ANR-10-SEGI-015 PRINCE Project 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.

References

1. Abdalla, M., An, J.H., Bellare, M., Namprempre, C.: From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 418–433. Springer (Apr / May 2002)
2. Abdalla, M., Ben Hamouda, F., Pointcheval, D.: Tighter reductions for forward-secure signature schemes. In: PKC 2013. LNCS, Springer (2013), full version available from the webpage of the authors.
3. Abdalla, M., Fouque, P.A., Lyubashevsky, V., Tibouchi, M.: Tightly-secure signatures from lossy identification schemes. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 572–590. Springer (Apr 2012)

4. Bellare, M., Miner, S.K.: A forward-secure digital signature scheme. In: Wiener, M.J. (ed.) CRYPTO'99. LNCS, vol. 1666, pp. 431–448. Springer (Aug 1999)
5. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Ashby, V. (ed.) ACM CCS 93. pp. 62–73. ACM Press (Nov 1993)
6. Cachin, C., Micali, S., Stadler, M.: Computationally private information retrieval with polylogarithmic communication. In: Stern, J. (ed.) EUROCRYPT'99. LNCS, vol. 1592, pp. 402–414. Springer (May 1999)
7. ECRYPT II yearly report on algorithms and keysizes (2011)
8. Feige, U., Fiat, A., Shamir, A.: Zero-knowledge proofs of identity. Journal of Cryptology 1(2), 77–94 (1988)
9. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer (Aug 1987)
10. Garg, S., Bhaskar, R., Lokam, S.V.: Improved bounds on security reductions for discrete log based signatures. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 93–107. Springer (Aug 2008)
11. Goldwasser, S., Micali, S., Rivest, R.L.: A "paradoxical" solution to the signature problem (abstract) (impromptu talk). In: Blakley, G.R., Chaum, D. (eds.) CRYPTO'84. LNCS, vol. 196, p. 467. Springer (Aug 1985)
12. Guillou, L.C., Quisquater, J.J.: A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In: Goldwasser, S. (ed.) CRYPTO'88. LNCS, vol. 403, pp. 216–231. Springer (Aug 1990)
13. Hohenberger, S., Waters, B.: Short and stateless signatures from the RSA assumption. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 654–670. Springer (Aug 2009)
14. Impagliazzo, R., Naor, M.: Efficient cryptographic schemes provably as secure as subset sum. Journal of Cryptology 9(4), 199–216 (1996)
15. Itkis, G., Reyzin, L.: Forward-secure signatures with optimal signing and verifying. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 332–354. Springer (Aug 2001)
16. Kakvi, S.A., Kiltz, E.: Optimal security proofs for full domain hash, revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 537–553. Springer (Apr 2012)
17. Katz, J., Wang, N.: Efficiency improvements for signature schemes with tight security reductions. In: Jajodia, S., Atluri, V., Jaeger, T. (eds.) ACM CCS 03. pp. 155–164. ACM Press (Oct 2003)
18. Lyubashevsky, V., Micciancio, D.: Generalized compact Knapsacks are collision resistant. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006, Part II. LNCS, vol. 4052, pp. 144–155. Springer (Jul 2006)
19. Malkin, T., Micciancio, D., Miner, S.K.: Efficient generic forward-secure signatures with an unbounded number of time periods. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 400–417. Springer (Apr / May 2002)
20. Micali, S.: A secure and efficient digital signature algorithm. Technical Memo MIT/LCS/TM-501b, Massachusetts Institute of Technology, Laboratory for Computer Science (Apr 1994)
21. Micali, S., Reyzin, L.: Improving the exact security of digital signature schemes. In: Baumgart, R. (ed.) CQRE'99. LNCS, vol. 1740, pp. 167–182. Springer (Nov / Dec 1999)
22. Micali, S., Reyzin, L.: Improving the exact security of digital signature schemes. Journal of Cryptology 15(1), 1–18 (2002), full version of [21]

23. Micciancio, D., Mol, P.: Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 465–484. Springer (Aug 2011)
24. Ohta, K., Okamoto, T.: A modification of the Fiat-Shamir scheme. In: Goldwasser, S. (ed.) CRYPTO'88. LNCS, vol. 403, pp. 232–243. Springer (Aug 1990)
25. Ong, H., Schnorr, C.P.: Fast signature generation with a Fiat-Shamir-like scheme. In: Damgård, I. (ed.) EUROCRYPT'90. LNCS, vol. 473, pp. 432–440. Springer (May 1990)
26. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT'99. LNCS, vol. 1592, pp. 223–238. Springer (May 1999)
27. Paillier, P., Vergnaud, D.: Discrete-log-based signatures may not be equivalent to discrete log. In: Roy, B.K. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 1–20. Springer (Dec 2005)
28. Patel, S., Sundaram, G.S.: An efficient discrete log pseudo random generator. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 304–317. Springer (Aug 1998)
29. Peikert, C., Rosen, A.: Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 145–166. Springer (Mar 2006)
30. Schnorr, C.P.: Efficient identification and signatures for smart cards (abstract) (rump session). In: Quisquater, J.J., Vandewalle, J. (eds.) EUROCRYPT'89. LNCS, vol. 434, pp. 688–689. Springer (Apr 1990)
31. Seurin, Y.: On the exact security of schnorr-type signatures in the random oracle model. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 554–571. Springer (Apr 2012)
32. van Oorschot, P.C., Wiener, M.J.: On Diffie-Hellman key agreement with short exponents. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 332–343. Springer (May 1996)