This extended abstract appeared in Quisquater Festschrif
D. Naccache Ed. Springer-Verlag, LNCS 6805, pages 143–155.

Autotomic Signatures

David Naccache and David Pointcheval

École normale supérieure
Département d'informatique, Groupe de cryptographie
45, rue d'Ulm, F-75230 Paris CEDEX 05, France
`david.{naccache,pointcheval}@ens.fr`

Abstract. Digital signature security is classically defined as an interaction between a signer $\mathcal{S}_{sk}$, a verifier $\mathcal{V}_{pk}$ and an attacker $\mathcal{A}$. $\mathcal{A}$ submits adaptively to $\mathcal{S}_{sk}$ a sequence of messages $m_1, \ldots, m_q$ to which $\mathcal{S}_{sk}$ replies with the signatures $U = \{\sigma_1, \ldots, \sigma_q\}$. Given U, $\mathcal{A}$ attempts to produce a forgery, *i.e.* a pair (m', σ') such that $\mathcal{V}_{pk}(m', \sigma') = \texttt{true}$ and $\sigma' \notin U$.

The traditional approach consists in hardening $\mathcal{S}_{sk}$ against a large query bound q. Interestingly, this is *one specific way* to prevent $\mathcal{A}$ from winning the forgery game. This work explores an alternative option.

Rather than hardening $\mathcal{S}_{sk}$, we weaken $\mathcal{A}$ by preventing him from influencing $\mathcal{S}_{sk}$'s input: upon receiving m_i, $\mathcal{S}_{sk}$ will generate a fresh ephemeral signature key-pair (sk_i, pk_i), use sk_i to sign m_i, erase sk_i, and output the signature and a certificate on pk_i computed using the long-term key sk. In other words, $\mathcal{S}_{sk}$ will only use his permanent secret sk to sign inputs which are *beyond $\mathcal{A}$'s control* (namely, freshly generated public-keys). As the sk_i are ephemeral, $q = 1$ by construction.

We show that this paradigm, called *autotomic signatures*, transforms weakly secure signature schemes (secure against generic attacks only) into strongly secure ones (secure against adaptively chosen-message attacks).

As a by-product of our analysis, we show that blending public key information with the signed message can significantly increase security.

1 Introduction

The security of cryptographic signatures is traditionally modeled as an interaction between an attacker $\mathcal{A}$, a signer $\mathcal{S}$ and a verifier $\mathcal{V}$. $\mathcal{A}$ adaptively submits to $\mathcal{S}$ a sequence of messages $m_1, \ldots, m_q$ to which $\mathcal{S}$ replies with the (corresponding) signatures $\sigma_1, \ldots, \sigma_q$.

As the interaction ceases, $\mathcal{A}$ attempts to produce a forgery (m', σ') such that:[1]

$$\mathcal{V}_{pk}(m', \sigma') = \texttt{true} \quad \text{and} \quad \sigma' \notin \{\sigma_1, \ldots, \sigma_q\}$$

The traditional approach consists in endeavoring to harden $\mathcal{S}$ against a large query bound q. Namely, design $(\mathcal{S}, \mathcal{V})$ in a way allowing to increase q non-polynomially at wish.

Interestingly, this is only *one specific way* to prevent $\mathcal{A}$ from forging. This paper explores an alternative approach that prevents $\mathcal{A}$ from adaptively influencing $\mathcal{S}$'s input.

The idea is the following: To sign a message m_i, $\mathcal{S}$ will:

– Generate a fresh ephemeral signature key-pair (sk_i, pk_i)
– Use sk_i to sign m_i. Let $\sigma_i = \mathcal{S}_{sk_i}(m_i)$ be the corresponding signature.
– Erase sk_i and output (σ_i, pk_i, c_i) where $c_i = \mathcal{S}_{sk}(pk_i)$ is a certificate on pk_i.

The verifier will check that:

$$\mathcal{V}_{pk_i}(m, \sigma_i) = \texttt{true} \quad \text{and} \quad \mathcal{V}_{pk}(pk_i, c_i) = \texttt{true}$$

In other words, $\mathcal{S}$ will only use sk to sign "sterilized" input which is *beyond $\mathcal{A}$'s control* (freshly generated public-keys). As each ephemeral secret key sk_i is used only once, $q = 1$ by construction.

We show that this paradigm, called *autotomic signatures* suffices to transform weakly secure signature schemes (generic attack secure) into strongly secure ones (adaptively chosen-message secure).

Autotomy[2] (or self amputation) is the act whereby an animal ($\mathcal{S}$) severs an appendage (sk_i, pk_i) as a self-defense mechanism designated to elude a predator's ($\mathcal{A}$) grasp. The lost body part being regenerated later (sk_{i+1}, pk_{i+1}). Typically, lizards and geckoes captured by the tail will shed part of the tail and thus be able to flee. Hence the name chosen for this paradigm.

[1] As is customary, we denote by (sk, pk) the key-pair of $\mathcal{S}$.
[2] *Αυτοτομία* from *αυτός* (*autós* = self) and *τέμνειν* (*temnein* = to cut).

2 Formal Framework

In [10], Goldwasser, Micali and Rivest defined *existential unforgeability against adaptive chosen-message attacks* (EUF-CMA) for digital signatures. EUF-CMA is today's *de facto* standard definition for digital signature security.

[10] define a scenario in which the adversary $\mathcal{A}$, given a target user's public key *pk*, is asked to produce a new message + signature pair (forgery) valid with respect to *pk*. For doing so, $\mathcal{A}$ is granted access to a signature oracle $\mathcal{S}$ (in practice, the legitimate signer himself) responding with signatures to challenge messages of $\mathcal{A}$'s choosing.

Formally, given a security parameter k, a signature scheme is a set of three algorithms $(\mathcal{K}, \mathcal{S}, \mathcal{V})$:

– A probabilistic *key generation algorithm* $\mathcal{K}$, which, on input 1^k, outputs a pair (pk, sk) of matching public and private keys.
– A (generally probabilistic) *signing algorithm* $\mathcal{S}$, which receives a message m and *sk*, and outputs a signature $\sigma = \mathcal{S}_{sk}(m)$.
– A (generally deterministic) *verification algorithm* $\mathcal{V}$, which receives a candidate signature σ, a message m and a public key *pk* and returns a bit $\mathcal{V}_{pk}(m, \sigma)$ representing the validity of σ as a signature of m with respect to *pk* *i.e.*:

$$\sigma = \mathcal{S}_{sk}(m) \quad \Rightarrow \quad \mathcal{V}_{pk}(m, \sigma) = \texttt{true}$$

Attacks against signature schemes are traditionally classified according to $\mathcal{A}$'s goals and resources. The most prevalent goals in the literature are:

Total Break	=	$\mathcal{A}$ outputs *sk*.
Universal Forgery	=	$\mathcal{A}$ signs any message.
Selective Forgery	=	$\mathcal{A}$ signs a message chosen before *pk* is known.
Existential Forgery	=	$\mathcal{A}$ signs some message.

It is easy to see that:

$$\text{Total Break} \Rightarrow \text{Universal Forgery} \Rightarrow \text{Selective Forgery} \Rightarrow \text{Existential Forgery}$$

Remark 1. In the above, $\mathcal{A}$ should be read as "an efficient $\mathcal{A}$ succeeding with a significant probability". The terms "efficiency" and "significant probability" admit various formal definitions.

A signature scheme capable of preventing any adversary from generating existential forgeries is called existentially unforgeable (EUF).

Literature abounds on the *resources* at $\mathcal{A}$'s command. In this paper we focus on two specific resource settings: *no-message attacks* and *known-message attacks*. In a no-message attack the adversary uses only *pk* to produce the forgery, whereas in known-message attacks $\mathcal{A}$ is given a list of valid message + signature pairs to work with.

Banishing existential forgeries (*i.e.* EUF-security), removes the lesser form of threat and hence offers the highest form of security (in other words, $\mathcal{A}$ is unable to sign even one, very specific, new message).

Nonetheless, a stronger security notion termed strong unforgeability (SUF)[3] [21], was recently defined for probabilistic signature schemes. In a SUF-secure scheme $\mathcal{A}$ cannot even create *new* signatures of messages *legitimately signed* by $\mathcal{S}$.

Finally, constraints on $\mathcal{A}$'s *modus operandi* categorize known message attacks into three subcases:

[3] Also called *non-malleability*.

Attack		Message choice process
Random-Message Attacks	RMA	by $\mathcal{S}$ at random
Generic Chosen-Message Attacks	GCMA	by $\mathcal{A}$ independently of pk and $\mathcal{S}$'s previous answers
Adaptive Chosen-Message Attacks	CMA	by $\mathcal{A}$ as a function of pk and $\mathcal{S}$'s previous answers

Clearly, CMA is the most stringent *modus operandi* and CMA $\Rightarrow$ GCMA $\Rightarrow$ RMA.

Restricting $\mathcal{A}$ to one signature requires even weaker security definitions integrating one-timeness. We thus extend GCMA and CMA as follows:

Definition 2 (OTCMA). One-Time-CMA, denoted OTCMA, is a CMA where $\mathcal{A}$ is limited to a single signature query.

Definition 3 (OTGCMA). One-Time-GCMA, denoted OTGCMA, is a GCMA where $\mathcal{A}$ is limited to a single signature query.

2.1 Connection to Related Work

The incorporation of randomness in signed strings is commonly used to upgrade signature security (*e.g.* PFDH [5]) or achieve tighter reduction bounds (*e.g.* PSS [2, 14]). Randomness generally allows the proof's simulator to generate signatures that do not help the adversary. Autotomic signatures can be regarded as a variant of this general design methodology where the ephemeral public-key acts as a randomness source. Note as well that two-stage signatures using ephemeral key-pairs were already used in the past. Most notably by Groth for designing provably secure group signatures [11, 3] (see as well [1]). The main contributions of this paper are hence a formalization of the concept, the proof of generic results and the illustration of the construction with concrete instances.

3 Autotomic Signatures

Consider two signature schemes $\Sigma^0 = (\mathcal{K}^0, \mathcal{S}^0, \mathcal{V}^0)$ and $\Sigma^1 = (\mathcal{K}^1, \mathcal{S}^1, \mathcal{V}^1)$ where Σ^1's message space contains Σ^0's public key space.

The autotomic scheme $\Sigma = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ is formally defined as follows:

- $\mathcal{K}$ runs $(sk^0, pk^0) \leftarrow \mathcal{K}^0$, and sets $(sk, pk) = (sk^0, pk^0)$.
- $\mathcal{S}_{sk}(m)$:
  - runs $\mathcal{K}^1$ to get an ephemeral key-pair (sk^1, pk^1).
  - certifies (signs) pk^1 using $\mathcal{S}^0_{sk}$, signs m using $\mathcal{S}^1_{sk^1}$ and erases sk^1.

$$c \leftarrow \mathcal{S}^0_{sk}(pk^1) \quad \text{and} \quad \sigma \leftarrow \mathcal{S}^1_{sk^1}(m)$$

  - The autotomic signature of m is defined as (pk^1, σ, c).
- $\mathcal{V}_{pk}(m, \sigma)$ verifies the signature by checking that

$$\mathcal{V}^1_{pk^1}(m, \sigma) = \texttt{true} \quad \text{and} \quad \mathcal{V}^0_{pk}(pk^1, c) = \texttt{true}$$

The following theorem, states that the autotomic combination of two weak signature schemes results in a strong signature scheme:

Theorem 4. *If Σ^0 is EUF-GCMA secure, and if Σ^1 is EUF-OTGCMA secure, then Σ is EUF-CMA secure.*

Proof. The proof proceeds in two steps: we reduce an adversary attacking Σ into either an adversary against Σ^0 (inner scheme forgery) or an adversary against Σ^1 (outer scheme forgery).

Consider an adversary $\mathcal{A}$ contradicting Σ's EUF-CMA security, given $pk = pk^0$, and q oracle accesses to Σ. We denote the queries by $m_1, \ldots, m_q$ and their answers by (pk^1_i, σ_i, c_i). $\mathcal{A}$ succeeded in generating a new message m and a corresponding forgery (pk', σ, c').

If such a forgery is possible with probability $\geq \varepsilon$, two events may occur:

***$pk' \notin \{pk^1_1, \dots pk^1_q\}$* with probability $\geq \varepsilon/2$.** In other words, $(m, (pk', \sigma, c'))$ is a an inner scheme forgery. This implies the existence of an adversary contradicting Σ^0's EUF-GCMA security constructed as follows:

Generate q public-private key-pairs (sk'_i, pk'_i) for Σ_1 and ask (non-adaptively) $\mathcal{S}^0$ for the signatures of all the pk'_i (let σ_i denote these signatures). This takes place before $\mathcal{A}$ sees the public key pk.

Ask the public key pk^0 corresponding to Σ^0 and set it as the public key of Σ (*i.e.* $pk = pk^0$). For each signature query m_i by $\mathcal{A}$, we take a pair (sk'_i, pk'_i) from the above queries list to Σ_1, sign m_i under sk'_i using $\mathcal{S}_1$ to get σ'_i, and output $(pk'_i, \sigma_i, \sigma'_i)$.

Then, $\mathcal{A}$ outputs an inner scheme forgery $(m, (pk', \sigma, c'))$, for a new pk':

$$\mathcal{V}^0(pk^0, pk', \sigma) = \texttt{true} \quad \text{and} \quad \mathcal{V}^1(pk', m, c') = \texttt{true}$$

Thus (pk', σ) is an existential forgery of Σ^0.

***$pk' \in \{pk^1_1, \dots pk^1_q\}$* with probability $\geq \varepsilon/2$.** Here $(m, (pk', \sigma, \sigma'))$ is an outer scheme forgery.

This case immediately leads to an adversary against EUF-OTGCMA of Σ^1. We will start the security game of Σ^1 later. We first generate a pair of public-private keys (sk^0, pk^0) for Σ^0 and randomly select an index k such that $1 \leq k \leq q$.

- *For the i-th signing query* m_i $(i \neq k)$, we generate a public-private key-pair (sk'_i, pk'_i) for Σ^1 and compute $\sigma_i = \mathcal{S}_0(sk^0, pk'_i)$, and $\sigma'_i = \mathcal{S}_1(sk'_i, m_i)$. The signature of m_i is thus the triple $(pk'_i, \sigma_i, \sigma'_i)$.
- *For the k-th signing query* m_k, we start the security game against Σ^1: we ask the signature σ'_k of m_k, before getting the public key pk'_k. As we get pk'_k, we can certify it $\sigma_k = \mathcal{S}_0(sk^0, pk'_k)$. The signature of m_k is thus the triple $(pk'_k, \sigma_k, \sigma'_k)$.

Finally, $\mathcal{A}$ outputs an outer scheme forgery $(m, (pk', \sigma, \sigma'))$, where $pk' = pk'_i$ for some i:

$$\mathcal{V}^0(pk^0, pk'_i, \sigma) = \texttt{true} \quad \text{and} \quad \mathcal{V}^1(pk'_i, m, \sigma') = \texttt{true}$$

with probability $1/q$, $i = k$, and then (m, σ') is an existential forgery of Σ^1 under the challenge key pk'_k. □

The above also applies to strong unforgeability:

Theorem 5. *If Σ^0 is SUF-GCMA secure, and Σ^1 is SUF-OTGCMA secure, then Σ is SUF-CMA secure.*

Proof. Consider an adversary $\mathcal{A}$ against Σ's SUF-CMA security: given a public key $pk = pk^0$, and q accesses to the signing oracle Σ, on messages m_i, with answers $(pk'_i, \sigma_i, \sigma'_i)$, $\mathcal{A}$ is able to generate a new signature (pk', σ, σ') for a message $m = m_i$ for some index i (if this is a new message, we can apply the previous proof).

Two situations can arise, since we assume that such a forgery occurs with probability $\geq \varepsilon$:

***$pk' \notin \{pk^1_1, \dots pk^1_q\}$* with probability $\geq \varepsilon/2$.** This case immediately leads to an adversary against EUF-GCMA of Σ^0.

Then, the adversary outputs an inner scheme forgery $(m, (pk', \sigma, \sigma'))$, for a new pk':

$$\mathcal{V}^0(pk^0, pk', \sigma) = \texttt{true} \quad \text{and} \quad \mathcal{V}^1(pk', m, \sigma') = \texttt{true}$$

thus (pk', σ) is a strong forgery of Σ^0.

***$pk' \notin \{pk_1^1, \ldots pk_q^1\}$* with probability $\geq \varepsilon/2$.** Let i be the index for which $pk' = pk_i'$:

- either $\forall i,\ (m, pk') \neq (m_i, pk_i)$, then at least (pk', σ) or (m, σ') is a strong forgery;
- or $\exists j,\ (m, pk') = (m_j, pk_j')$, then at least
  - either $\sigma \neq \sigma_i$, and thus (pk', σ) is a strong forgery of Σ^0,
  - or $\sigma' \neq \sigma_i'$, and thus (m, σ') is a strong forgery of Σ^1.

□

4 Concrete Instantiations

4.1 Autotomic Chameleon Signatures

[15] introduces the concept of *Chameleon Signatures* using *Chameleon Hash Functions* (CHF). A CHF can be seen as one-time signature: A CHF H uses a public key pk to compute $h = H_{pk}(m, r)$ and satisfies the following properties:

Collision-Resistance: Given pk it is hard to find $(m_1, r_1) \neq (m_2, r_2)$ such that

$$H_{pk}(m_1, r_1) = H_{pk}(m_2, r_2)$$

Trapdoor: Given a trapdoor information sk, it is easy to compute a second pre-image with a specific prefix. Formally, given $(sk, (m_1, r_1), m_2)$, one can compute r_2 such that $H_{pk}(m_1, r_1) = H_{pk}(m_2, r_2)$.

Uniformity: When r is uniformly distributed, $H_{pk}(m, r)$ perfectly (or computationally) hides the message m.

To see H as a signature scheme $(\mathcal{K}, \mathcal{S}, \mathcal{V})$, we define:

- $\mathcal{K}$ runs the CHF's key generation process, gets sk_h and pk_h, chooses a random message m_1, a random number r_1, and computes $h = H_{pk_h}(m_1, r_1)$. It sets $sk = (sk_h, m_1, r_1)$ and $pk = (pk_h, h)$.
- Given a message m, $\mathcal{S}$ gets from sk_h and (m_1, r_1), the number r such that $h = H_{pk_h}(m_1, r_1) = H_{pk_h}(m, r)$. The signature simply consists of r, which satisfies $h = H_{pk_h}(m, r)$.
- The verification of the equality $h = H_{pk_h}(m, r)$ is the verification algorithm $\mathcal{V}$.

Theorem 6. *If H is a* CHF *then the above signature scheme is* SUF-OTGCMA *secure.*

Proof. We are given a public key pk_h. $\mathcal{A}$ submits first its unique query m to the signing oracle, before seeing the signing public key. We choose a random r,compute $h = H_{pk_h}(m, r)$ and set $pk = (pk_h, h)$. Given the uniformity property, h is independent of m, and thus the key pk is uniformly distributed among the public keys.

The signature of m is thus r, since $h = H_{pk_h}(m, r)$.

If the adversary is able to forge a new (m', r') pair, then $h = H_{pk_h}(m, r) = H_{pk_h}(m', r')$, which is a collision, for a given public key pk_h only, which is hard to produce[4].

Hence, it appears that this can be combined with any SUF-GCMA secure signature scheme to get a SUF-CMA secure signature scheme.

[4] Under the collision-resistance assumption of the hash function family

4.2 DL-Based Instantiation

Consider the following DLP-based example:

- The functions apply in a group $\mathbb{G}$ of prime order q, where g is a generator;
- The keys are: $sk = x \overset{R}{\leftarrow} \mathbb{Z}_q$, $pk = y = g^x$
- The hash function is defined by $h = H_{pk}(m, r) = g^m y^r$

It is clear that finding a collision solves the DLP of y in basis g, but given the discrete logarithm x, as well as $h = g^{m_1} y^{r_1}$, and a message m_2, it is clear that $r_2 = r_1 + (m_1 - m_2)/x \bmod q$ makes that $H_{pk}(m_1, r_1) = H_{pk}(m_2, r_2)$. Uniformity is perfect, since y is a generator, and thus y^r for a random r perfectly hides m.

Theorem 7. *The above signature scheme is $\mathsf{SUF\text{-}OTGCMA}$ secure, under the discrete logarithm assumption.*

4.3 RSA-Based Instantiation

Consider the following example based on the RSA problem:

- The functions apply in a $\mathbb{Z}_n$ for an RSA modulus n, with exponent e;
- The keys are: $sk = x \overset{R}{\leftarrow} \mathbb{Z}_n$, $pk = y = x^e \bmod n$
- The hash function is defined by $h = H_{pk}(m, r) = m^e y^r \bmod n$

It is clear that finding a collision leads to

$$h = {m_1}^e y^{r_1} = {m_2}^e y^{r_2} \quad \Rightarrow \quad y^{r_2 - r_1} = (m_1/m_2)^e \bmod n.$$

If $r_2 - r_1$ and e are co-prime, then we can extract the e-th root of y. For that to always hold, we need $e = n$.

Uniformity is statistical, as soon as y is of order large enough, which holds which overwhelming probability.

Theorem 8. *The above signature scheme is $\mathsf{SUF\text{-}OTGCMA}$ secure, under the $\mathsf{RSA}_{n,n}$ assumption.*

5 More Constructions

Since we have two one-time signatures, we now have to look for new signature schemes to combine with. The list of candidates is

- Schnorr's signature [20], which is already $\mathsf{SUF-CMA}$ in the random oracle model, under the DL assumption [17]. Without the random oracle, there is no hope to prove anything. Any conversion from an identification scheme using the Fiat-Shamir [8] paradigm will have the same problem (all the Schnorr's variants, such as DSA, and GQ signature [12]), because of the simulator of the zero-knowledge proof system;
- RSA signature [18], which is already $\mathsf{SUF-CMA}$ in the random oracle model, under the RSA assumption [2, 4]. Without the random oracle, it is existentially forgeable under no-message attacks. There is thus no hope to prove anything either;
- GHR signature [9], is a hash-then-invert like signature, also known as Full-Domain Hash, but by opposition to the RSA case, it can reach a minimal security level without random oracle, we thus focus on this signature scheme.

5.1 Gennaro-Halevi-Rabin Signatures

In 1999, Gennaro, Halevi and Rabin [9] proposed a signature scheme, which basically works as follows:

- The key generation process generates an RSA modulus n and a random $y \in \mathbb{Z}_n$. The public key consists of the pair (n, y), and the private key in the pair (p, q).
- The signing process, given a message m, first computes $e = H(m)$, so that it is prime to $\varphi(n)$ (we discuss that later). One then computes $d = e^{-1} \bmod \varphi(n)$, and the signature $s = y^d \bmod n$.
- To verify the signature s with respect to the message m and the public key (n, y), one simply checks whether $y = s^{H(m)} \bmod n$.

In [9], the authors prove the following security result:

Theorem 9. *The above signature scheme is SUF-CMA secure, under the Strong RSA assumption, in the random oracle model.*

They also reduced the random oracle requirement to a hash function that is division intractable, plus some randomness properties:

Definition 10 (Division-Intractability). A hash function H is said division-intractable if it is hard to find $x_1, \ldots, x_q$ and y such that $H(y)$ divides $\prod H(x_i)$.

However, if H behaves like a random function, Coron and Naccache [6] showed that a 1024-bit hash function is necessary for a security level similar to RSA-1024, contrarily to the 512-bit size suggested by the authors.

Modulus. To ascertain that $H(m)$ is invertible modulo $\varphi(n)$, we can choose a strong RSA modulus n (which means that $n = pq$, such that both $(p-1)/2$ and $(q-1)/2$ are prime numbers), and define $H(x) = 2h(x) + 1$ to make it odd. $H(x)$ is thus prime to $\varphi(n)$ with overwhelming probability. Then, we have to choose a hash function with an output length similar to the size of n, which implies a full-size exponentiation for the verification.

Alternative to Division-Intractability. Another possibility is to define $H(m, r) = h(m)||r$, where r is the smallest integer such that $H(m, r)$ is a prime number. This allows to use a very short exponent, at low cost. Indeed, Cramér's conjecture says that the bigger gap between 2 consecutive primes p and p' is $\log(p)^2$. As a consequence, with a 256-bit hash function h, 16 additional bits are enough to guarantee that a prime will appear in the interval. On average, the gap is $\log p$ only, and thus on average 64 primality tests are enough (since we can safely exclude even numbers). Such a function is division-intractable in the perfect sense, since such sequences do not exist (all the output numbers are distinct primes).

5.2 Resistance to Generic Attacks

Note that an efficient way to limit the impact of Coron-Naccache's attack is to use a hash function that takes both the message and the public key (or at least some variable part of it) as input. Since we just want to achieve GCMA security, the adversary cannot prepare the messages to be queried and then see the exponents it will obtain. The attack has thus to be generic: the adversary first chooses q messages, when it knows the public key, it learns the q exponents for which it gets the signatures. It then has to generate a message digest dividing the product of all theses exponents, which is very unlikely.

Description of our Scheme. We thus suggest the following modification, where the hash function H just needs to have some uniformity properties and output odd integers only:

- The key generation process first chooses a strong RSA modulus n and a random element $y \in \mathbb{Z}_n$. The public key consists of the pair (n, y), and the private key in the pair (p, q).
- The signing process, given a message m, first computes $e = H(n, m)$. One then computes $d = e^{-1} \bmod \varphi(n)$, and the signature $s = y^d \bmod n$.

– To verify the signature s with respect to the message m and the public key (n, y), one simply checks whether $y = s^{H(n,m)} \bmod n$.

Since we can limit the number of generic signature requests to say 2^{30}, the output length of the hash function can be reduced to 160 bits. For a 256-bit hash function, we can set the usage limit up to 2^{40}, according to [6].

Security Result. Let us now prove security under division intractability:

Theorem 11. *The above signature scheme is SUF-GCMA secure, under the Strong RSA assumption and the division-intractability of the function H.*

Proof. We are given a flexible RSA instance n, y. The adversary first submits its queries $m_1, \ldots, m_q$ to the signing oracle, before seeing the signing public key. We thus compute $e_i = H(n, m_i)$ for $i = 1, \ldots, q$, and then $Y = y^{\prod e_i} \bmod n$. The public key is set to $pk = (n, Y)$.

The signature of m_i is thus $s_i = y^{\prod_{j \neq i} e_i} = Y^{1/e_i} \bmod n$.

If the adversary is able to forge a new pair (m, s), then $s^e = Y = y^{\prod e_i} \bmod n$. Under the division intractability assumption, e does not divide $E = \prod e_i$, and thus there exist a, e' and E' such that $e = e'a$ and $E = E'a$, with $e' \neq 1$, and e' relatively prime to E'. Since we considered a strong RSA modulus n, with odd exponents, e' is relatively prime to $\varphi(n)$:

$$ue' + vE' = 1 \quad s^{e'} = Y = y^{E'}.$$

We set $X = s^v y^u$, then

$$X^{e'} = s^{e'v} y^{e'u} = y^{E'v} y^{e'u} = y \bmod n,$$

which solves the F-RSA problem.

Combined with the RSA-based chameleon hash function, we obtain a signature scheme that is SUF – CMA under the Strong RSA assumption.

5.3 A Complete Signature Scheme

We can now describe the complete signature scheme, which relies on the Flexible RSA problem (F-RSA), and a weaker division-intractability assumption:

– The key generation process chooses a strong RSA modulus n and a random element $y \in \mathbb{Z}_n$. The public key consists of the pair (n, y). The private key is the pair (p, q).

– The signing process, given a message m, first generates a random element $x \in \mathbb{Z}_n$, computes $z = x^n \bmod n$, chooses a random $r \in \mathbb{Z}_n$ and computes $h = z^r m^n \bmod n$.

  Let $e = H(n, h, z)$ and $d = e^{-1} \bmod \varphi(n)$. The signature consists of $s = y^d \bmod n$, and (z, r).

– To verify the signature (s, z, r) with respect to the message m and the public key (n, y), the verifier simply checks whether $y = s^{H(n,h,z)} \bmod n$, where $h = z^r m^n \bmod n$.

The assumption on the hash function is the following:

Definition 12 (Weak Division-Intractability). A hash function family H_k is q-weak division-intractable if it is hard to win the following game: the adversary chooses q elements $x_1, \ldots, x_q$, the defender chooses a random key k, and the adversary has to find y such that $H_k(y)$ divides $\prod H_k(x_i)$.

We can indeed see the additional input n to the hash function as a key.

5.4 Keyed Hash Functions

A crucial point raised in the previous analysis is the impact on security caused by including the public key (or at least a variable part of it) in the hash value. It has already been noticed for hash functions in general [19, 13], essentially to make the collision-resistance definition formal, since it defines a hash function family:

Definition 13 (Collision-Resistant Hash Functions (CRHF)). A family of functions H_k is *collision-resist* if, for a random key k as input, it is hard to find x and y such that $H_k(x) = H_k(y)$.

To compare the above notions of *division-intractability* and *weak division-intractability*, let us review the definition of Universal One-Way Hash Functions [16]:

Definition 14 (Universal One-Way Hash Functions (UOWHF)). A family of functions H_k is *universal one-way* if, after having chosen x, for a random key k, it is hard to find y such that $H_k(x) = H_k(y)$.

It is widely believed that the UOWHF assumption is much weaker than the CRHF assumption. One should then note that if the public key is inserted into the input of the hash value to be signed, in the case of generic attacks only, UOWHF are basically enough.

Note that the difference between the definitions of CRHF and UOWHF is similar to the difference between the definitions of Division-Intractability and Weak Division-Intractability. The latter is thus much weaker than the former.

Also, in the above scheme, the security result requires Weak Division-Intractability instead of Division-Intractability if a variable part of the public key is insert in the hash value. The entire public key would weaken the security result!

5.5 Weak Division-Intractability

To evaluate the size of the output to achieve Weak Division-Intractability, a similar evaluation to [6] should be performed. We believe that the attack is much more difficult in this case.

6 Acknowledgments

The authors are grateful to Damien Vergnaud and Nigel Smart for their useful comments and suggestions regarding this work.

References

1. G. Ateniese, J. Camenisch, S. Hohenberger, and B. de Medeiros. Practical group signatures without random oracles. Cryptology ePrint Archive, Report 2005/385, 2005. `http://eprint.iacr.org/`.
2. M. Bellare and P. Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In U. M. Maurer, editor, *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer, May 1996.
3. A. Boldyreva, M. Fischlin, A. Palacio, and B. Warinschi. A closer look at PKI: Security and efficiency. In T. Okamoto and X. Wang, editors, *PKC 2007: 10th International Conference on Theory and Practice of Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 458–475. Springer, Apr. 2007.
4. J.-S. Coron. On the exact security of full domain hash. In M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer, Aug. 2000.
5. J.-S. Coron. Optimal security proofs for PSS and other signature schemes. In L. R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 272–287. Springer, Apr. / May 2002.
6. J.-S. Coron and D. Naccache. Security analysis of the Gennaro-Halevi-Rabin signature scheme. In B. Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 91–101. Springer, May 2000.
7. R. Cramer and V. Shoup. Signature schemes based on the strong RSA assumption. In *ACM CCS 99: 6th Conference on Computer and Communications Security*, pages 46–51. ACM Press, Nov. 1999.

8. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, Aug. 1987.
9. R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In J. Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 123–139. Springer, May 1999.
10. S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, Apr. 1988.
11. J. Groth. Fully anonymous group signatures without random oracles. In K. Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 164–180. Springer, Dec. 2007.
12. L. C. Guillou and J.-J. Quisquater. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In S. Goldwasser, editor, *Advances in Cryptology – CRYPTO'88*, volume 403 of *Lecture Notes in Computer Science*, pages 216–231. Springer, Aug. 1990.
13. A. Joux. Can we settle cryptography's hash?, 2009. Invited talk at the ACNS '09 Conference.
14. J. Katz and N. Wang. Efficiency improvements for signature schemes with tight security reductions. In S. Jajodia, V. Atluri, and T. Jaeger, editors, *ACM CCS 03: 10th Conference on Computer and Communications Security*, pages 155–164. ACM Press, Oct. 2003.
15. H. Krawczyk and T. Rabin. Chameleon signatures. In *ISOC Network and Distributed System Security Symposium – NDSS 2000*. The Internet Society, Feb. 2000.
16. M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *21st Annual ACM Symposium on Theory of Computing*, pages 33–43. ACM Press, May 1989.
17. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.
18. R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signature and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978.
19. P. Rogaway and T. Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In B. K. Roy and W. Meier, editors, *Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 371–388. Springer, Feb. 2004.
20. C.-P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, Aug. 1990.
21. J. Stern, D. Pointcheval, J. Malone-Lee, and N. P. Smart. Flaws in applying proof methodologies to signature schemes. In M. Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 93–110. Springer, Aug. 2002.

A Standard Definitions

Definition 15 (DL). The Discrete Logarithm problem (DL) in base g in a group $\mathbb{G}$ of prime order q, denoted $\mathsf{DL}_{\mathbb{G},g}$, consists, given $y = g^x$, in computing $x \bmod q$.

The *DL assumption* conjectures the intractability of the DL problem.

Definition 16 (RSA). The RSA problem (RSA) with modulus n and exponent e, denoted $\mathsf{RSA}_{n,e}$, consists, given $y = x^e \bmod n$, in computing $x \bmod n$.

The *RSA assumption* conjectures the intractability of the RSA problem.

Definition 17 (F-RSA). The Flexible RSA problem (F-RSA) [7] with modulus n,denoted $\mathsf{F\text{-}RSA}_n$, consists, given $y \bmod n$, in computing a pair (e, x), for a prime exponent e, such that $y = x^e \bmod n$.

The *Strong RSA assumption* conjectures the intractability of the F-RSA problem.