Published T.-W. Kuo and S. Cruz-Lara, Eds, *4th Taiwanese-French Conference on Information Technology (TFIT 2008)*, pp. 322–327, Taipei, Taiwan, March 3–5, 2008.

Secure Design of RSA-based Cryptosystems[*]

Marc Joye
Thomson R&D France
Technology Group, Corporate Research, Security Laboratory
1 avenue de Belle Fontaine, 35576 Cesson-Sévigné Cedex, France
marc.joye@thomson.net – http://www.geocities.com/MarcJoye/

Abstract

The design of secure cryptosystems is of central importance in cryptography. Textbook RSA does not achieve the highest security level and so cannot be used in all contexts. This paper surveys several RSA-based proposals introduced so far and discusses their security. A special attention is paid to the recent TSS signature scheme.

1. Introduction

A digital signature is the digital counterpart of an handwritten signature. This is a useful cryptographic primitive as it provides the properties of authentication, integrity and non-repudiation.

The most well-known signature scheme is the *textbook RSA signature*. On input a security parameter k and public exponent e, the key generation consists (i) in generating two large primes p and q such that $\gcd(p-1, e) = \gcd(q-1, e) = 1$ to form k-bit RSA modulus $N = pq$, and (ii) in computing $d = e^{-1} \bmod \phi(N)$ where $\phi(N) = (p-1)(q-1)$. The public key is $\mathsf{pk} = \{N, e\}$ and the private key is $\mathsf{sk} = \{d\}$. The signature on a message m is given by $\sigma = m^d \bmod N$. The validity of σ can then be publicly checked by verifying whether $\sigma^e \equiv m \pmod N$.

Unfortunately, textbook RSA signature is subject to forgeries. It is easy to see that a random $r \in \mathbb{Z}_N$ is a valid signature on "message" $m = r^e \bmod N$. More dramatically, due to its multiplicative property, textbook RSA signature is also subject to selective forgeries. The signature on a *chosen* message m can be obtained as follows:

- choose a random $r \in \mathbb{Z}_N$ and compute $m' = m/r^e \bmod N$;
- obtain the signature σ' on message m': $\sigma' = m'^d \bmod N$;
- recover the signature σ on chosen message m as $\sigma = \sigma' r \bmod N$.

Computing e-th roots modulo N is problem conjectured difficult — this is the RSA assumption. However, this problem becomes easy when access is given to an oracle returning the signature on *chosen* messages: textbook RSA signatures are universally forgeable under chosen-message attacks.

The previous forgery illustrates the need to precisely define what means security: this requires to define a security model and the adversary's resources. In the next sections, we will define the security notions for signature schemes and for encryption schemes. We will explain how to prove the security of several RSA-based schemes in certain security models under some cryptographic assumptions.

2. Signature Schemes

Formally, a *digital signature scheme* is a set of 3 algorithms:

1. Key generation: on input a security parameter, it produces a matching pair $(\mathsf{pk}, \mathsf{sk})$ of public key and private key;
2. Signing: on input signing key sk, message m (and optionally random r), it outputs the signature $\sigma = \mathscr{S}(\mathsf{sk}, m\,[, r])$;
3. Verification: on input verification key pk and signature σ (most verification algorithms also require message m), it returns $\mathscr{V}(\mathsf{pk}, \sigma\,[, m]) = 0$ or 1 (1 meaning that the signature is valid and 0 that it is not).

A *security notion* is a pair (security goal, attack scenario). For signature schemes, the easiest goal is to produce an existential forgery and the strongest attack scenario (in the black-box model) is to give access to a chosen-message

[*]Invited paper.

oracle. As a consequence, the highest security level is EUF-CMA, namely *existential unforgeability against chosen-message attacks*.

Basically, the idea behind *provable* security (or reductionist security [8]) is to prove that a scheme is secure by exhibiting a so-called reduction that uses a chosen-message attacker against the signature scheme, in order to solve a hard cryptographic problem. In the *standard model*, the attacker has access to a signing oracle that is simulated by the reduction, answering the signature queries on chosen messages, and receiving eventually a signature forgery.

Two classes of provably secure signature schemes can be distinguished. The first class proposes reductions that are said *loose*, as they can turn an attacker into an algorithm solving a cryptographic problem, but whose ratio $\rho =$ running time/success probability is far greater than the one required by the attacker to produce a forgery. Hence, this kind of reduction only proves the security asymptotically. The second class of provable signature schemes features so-called *tight* reductions, using the attacker to solve the cryptographic problem with roughly the same ratio ρ.

There are just a handful of practical RSA-based signature schemes with a security reduction in the standard model, and most of them rely on strong RSA assumption.

Being given a safe RSA modulus N (that is, an RSA modulus $N = pq$ with $p = 2p' + 1$ and $q = 2q' + 1$ and where p, q, p', q' are prime) and an element $y \in \mathbb{Z}_N^*$, the *flexible* RSA *problem* is defined as finding an element $x \in \mathbb{Z}_N^*$ and an integer $e > 1$ such that

$$x^e \equiv y \pmod{N} \ .$$

The *strong* RSA *assumption* conjectures that there is no attacker that can (τ, ϵ)-solve the flexible RSA problem (i.e., with success probability smaller than ϵ and running time bounded by τ) with τ polynomial and ϵ non-negligible. The difference with the (standard) RSA *assumption* is that (i) the attacker is free to choose the value of exponent e (provided it is > 1) and (ii) RSA modulus is supposed to be safe.

Below, we review several provably secure signature schemes. We begin with the Gennaro-Halevi-Rabin (GHR) signature scheme [7] the security of which relies on the strong RSA assumption, in the standard model. We then describe the full-domain-hash (FDH) signature scheme [2] so as to introduce the random oracle model. Finally, we present TSS [5] which a tight security reduction under the strong RSA assumption without random oracles. In order to illustrate the simulation/reduction paradigm, we provide a detailed security proof of this latter scheme.

2.1. GHR signature scheme

In the original description, the GHR scheme requires a hash function $\mathcal{H}$ which is *division-intractable* (see [7] for a precise definition). The easiest way to achieve this additional property is to define $\mathcal{H}$ as a hash function that maps bit-strings to prime numbers.

Key generation The public key is $\mathsf{pk} = \{N, u\}$ where $N = (2p' + 1)(2q' + 1)$ is a safe RSA modulus and u is a random element in $\mathbb{Z}_n^*$. The private key is $\mathsf{sk} = \{p', q'\}$.

Signing Let $\mathcal{M}$ denote the set of messages. To sign message $m \in \mathcal{M}$, compute

$$\sigma = u^{c^{-1} \bmod p'q'} \bmod N \quad \text{where } c = \mathcal{H}(m)$$

for a division-intractable hash function $\mathcal{H} : \mathcal{M} \to \{0,1\}^{\ell_h}$.

Verification Signature σ on message $m \in \mathcal{M}$ is valid if and only if $\sigma^{\mathcal{H}(m)} \equiv u \pmod{N}$.

Figure 1. GHR signature scheme.

The GHR signature scheme as above has a *loose* security reduction to the flexible RSA problem. However, in [7], the authors also propose techniques to achieve tightness in their signature scheme. Basically, their idea is to make use of a chameleon hash function [9]. Let P be an ℓ_p-bit prime, let Q be an ℓ_q-bit prime divisor of $P - 1$, and let $\langle g \rangle$ denote the cyclic subgroup generated by an element $g \in \mathbb{Z}_P^*$ of order Q. They so obtain a scheme such that an attacker against it can be used to solve either the discrete logarithm problem in subgroup $\langle g \rangle$ or the flexible RSA problem modulo N, with roughly the same success probability and running time.

2.2. RSA-FDH signature scheme

There exists no RSA-based signature scheme whose security solely relies on the RSA assumption (note that GHR requires the strong RSA assumption). Very recently, Paillier pointed out an impossibility result for a [wide] class of RSA-based signatures [10], implying that devising such a scheme is hardly feasible. It is however possible to prove the security in some idealized worlds.

One such idealization, referred to as the *random oracle model* [2, 4], consists in considering the output of a hash function as a perfect pseudo-random number generator. Within the random oracle model, provided that input message m is first hashed with a *full-domain hash function* (i.e., a hash function that maps bit-strings to the full set of integers modulo N) before entering the RSA primitive, then the resulting signature (called an RSA-FDH signature) attains the EUF-CMA security level.

Key generation The public key is $\mathsf{pk} = \{N, e\}$ with $N = pq$, and the private key is $\mathsf{sk} = \{d\}$ with $d = e^{-1} \bmod \phi(N)$.

Signing The signature on a message $m \in \mathcal{M}$ is given by

$$\sigma = \mathcal{H}(m)^d \bmod N$$

where $\mathcal{H} : \mathcal{M} \to \mathbb{Z}_N$ is a full-domain hash function.

Verification Signature σ on message $m \in \mathcal{M}$ is valid if and only if $\sigma^e \equiv \mathcal{H}(m) \pmod N$.

Figure 2. RSA-FDH signature scheme.

When proving the security in the random oracle model, the attacker gets in addition access to a hash oracle, also simulated by the reduction.

2.3. TSS signature scheme

The GHR scheme requires the use of a hash function that maps to prime numbers (or at least, that is division-intractable) but has a tight security reduction to the SRSA in the standard model when using a chameleon function. Our goal, when designing TSS [5], was to combine the practicality of the RSA-FDH scheme with the tightness of the GHR scheme, without random oracles.

Intuitively, there are two ideas behind our scheme. First, we use a fresh, prime exponent c in each signature generation and give a c^{th} root modulo a safe RSA modulus n as part of the signature. Further, we make prime c free of any particular relation (except its size), in order to allow the use of fast prime generation algorithms. The second idea is, as in the GHR scheme, to use a chameleon function in order to tighten the security reduction. In order to base the security on the strong RSA assumption, we define a second safe RSA modulus N and make use of an RSA-type chameleon function. The resulting construction is depicted in Fig. 3.

The security of TSS is *tightly* related the strong RSA assumption. That is, given an ℓ_n-bit safe RSA modulus $\hat{n}$ and a random element $\hat{y} \in \mathbb{Z}_{\hat{n}}^*$, the problem consists in finding a pair $(\hat{x}, \hat{e}) \in \mathbb{Z}_{\hat{n}}^* \times \mathbb{Z}_{>1}$ satisfying $\hat{y} \equiv \hat{x}^{\hat{e}} \pmod{\hat{n}}$. More formally, we have:

Theorem 1 *Suppose that the flexible* RSA *problem is* (τ, ϵ)*-hard. Then, for any* q_s *signature queries, the TSS signature scheme is* $(\tau_{\mathcal{A}}, q_s, \epsilon_{\mathcal{A}})$*-secure in the sense of* sEUF-CMA*, where*

$$\epsilon \geq \frac{\epsilon_{\mathcal{A}}}{2} \text{ and } \tau \lesssim \tau_{\mathcal{A}} + \mathrm{O}\big({\ell_n}^5 + q_s\, {\ell_n}^3 \max(\log q_s, \ell_n)\big) .$$

Key generation On input security parameters ℓ_n and ℓ_m:

- choose an (odd) $(\ell_m + 1)$-bit prime E;
- generate two random ℓ_n-bit safe RSA moduli $n = (2p'+1)(2q'+1)$ and $N = (2P'+1)(2Q'+1)$ such that $\gcd(P'Q', E) = 1$;
- compute $D = E^{-1} \bmod 2P'Q'$;
- choose at random two elements $u \in \mathbb{Z}_n^*$ and $g \in \mathbb{Z}_N^*$.

The public key is $\mathsf{pk} = \{N, n, u, g, E\}$ and the private key is $\mathsf{sk} = \{p', q', D\}$.

Signing Let $m \in \{0,1\}^{\ell_m}$ be the message to be signed:

- choose a random prime c in $[(N+1)/2, N[$;
- compute $s = u^{c^{-1} \bmod 2p'q'} \bmod n$ and $r = (c\, g^{-(m+1)})^D \bmod N$.

The signature on m is $\sigma = (r, s) \in \mathbb{Z}_N^* \times \mathbb{Z}_n^*$.

Verification Let $\sigma = (r, s)$ be a putative signature on message $m \in \{0,1\}^{\ell_m}$. Then

(i) check that $(r, s) \in [0, N[\times [0, n[$;

(ii) check that $s^c \equiv u \pmod n$ where $c = g^{m+1}\, r^E \bmod N$.

If the two conditions hold then signature σ is accepted.

Figure 3. TSS signature scheme.

Remember that a signature scheme is existentially unforgeable under chosen-message attacks (EUF-CMA) if no adversary is able to produce a valid signature on a *new* message (i.e., a message that was not submitted to the signing oracle). A slightly stronger security notion is that of sEUF-CMA: a scheme is said *strongly* existentially unforgeable under chosen-message attacks if no adversary is able to produce a new valid pair of message/signature (note that the distinction between EUF and sEUF only makes sense for probabilistic signature schemes).

Proof [of Theorem 1] The proof is by contradiction. We assume that there exists a polynomial-time adversary $\mathcal{A}$ that is able to produce a weak existential forgery with non-negligible success probability $\epsilon_{\mathcal{A}}$ within time $\tau_{\mathcal{A}}$ after q_s

queries to a signing oracle. We then use $\mathcal{A}$ to (τ, ϵ)-solve the flexible RSA problem, i.e., to find a pair $(\hat{x}, \hat{e})$ on input challenge $(\hat{n}, \hat{y})$.

We toss a coin $b \in \{0, 1\}$ and run Simulation b defined as follows.

Simulation 0

- We let $n = \hat{n}$. We choose an (odd) $(\ell_m + 1)$-bit prime E. Next, we generate a random ℓ_n-bit safe RSA modulus $N = (2P' + 1)(2Q' + 1)$ such that $\gcd(P'Q', E) = 1$. We compute $D = E^{-1} \bmod 2P'Q'$. We choose a random element $g \in \mathbb{Z}_N^*$. Finally, for all $i \in \{1, \ldots, q_s\}$, we let c_i be a random prime in $[(N + 1)/2, N[$ and define
$$u = \hat{y}^{\prod_i c_i} \bmod n \ .$$
We create the public key $\mathsf{pk} = \{N, n, u, g, E\}$. It is easy to see that the key generation is perfectly simulated.

- When $\mathcal{A}$ requests the signature on a message $m_j \in \{0, 1\}^{\ell_m}$, for $j \in \{1, \ldots, q_s\}$, we simulate the signing oracle by computing
$$r_j = (c_j\, g^{-(m_j+1)})^D \bmod N \quad \text{and} \quad s_j = \hat{y}^{\prod_{i \neq j} c_i} \bmod n \ .$$
We return $\sigma_j = (r_j, s_j)$ as the signature on m_j. Here too, the simulation is perfect.

Simulation 1

- We let $N = \hat{n}$ and $g = \hat{y}$. We choose a random ℓ_n-bit safe RSA modulus $n = (2p' + 1)(2q' + 1)$ and an (odd) $(\ell_m + 1)$-bit prime E. (W.l.o.g., we may assume that (odd) prime $E \in \mathbb{Z}_{2P'Q'}^*$ as otherwise we would have $E = P'$ or $E = Q'$, which yields the factorization of N.) Finally, we choose a random element $u \in \mathbb{Z}_n^*$.

  We create the public key $\mathsf{pk} = \{N, n, u, g, E\}$. The key generation is perfectly simulated.

- When $\mathcal{A}$ requests the signature on a message $m_j \in \{0, 1\}^{\ell_m}$, for $j \in \{1, \ldots, q_s\}$, we simulate the signing oracle as follows.

  1. We choose a random element $r_j \in \mathbb{Z}_N^*$ and define $c_j = g^{m_j+1}\, {r_j}^E \bmod N$;
  2. If c_j is not a prime lying in $[(N + 1)/2, N[$, then we go back to Step 1.

  Next, we compute $s_j = u^{c_j^{-1} \bmod 2p'q'} \bmod n$ and return $\sigma_j = (r_j, s_j)$ as the signature on m_j. The simulation is perfect.

Eventually, adversary $\mathcal{A}$ outputs with probability $\epsilon_{\mathcal{A}}$ a valid signature forgery $\sigma_* = (r_*, s_*) \in [0, N[\times [0, n[$ on a message $m_* \in \{0, 1\}^{\ell_m}$, with $(m_*, \sigma_*) \neq (m_i, \sigma_i)$ for all $i \in \{1, \ldots, q_s\}$. We compute $c_* := g^{m_*+1}\, {r_*}^E \bmod N$.

- If $c_* \neq c_j$ for all $j \in \{1, \ldots, q_s\}$, if $c_* > 1$, and if $b = 0$ (i.e., Simulation 0 was run) then it follows that $\gcd(c_*, \prod_i c_i) = 1$, since $c_* \in [2, N[$ and all c_i's are primes in set $[(N + 1)/2, N[$. Hence, from extended Euclidean algorithm, we get integers α and β s.t. $\alpha\, c_* + \beta \prod_i c_i = 1$. Therefore, noting that $\hat{y}^{\prod_i c_i} \equiv u \equiv {s_*}^{c_*} \pmod{n}$ and $n = \hat{n}$, we have
$$\hat{y} \equiv \hat{y}^{\alpha\, c_* + \beta \prod_i c_i} \equiv \left(\hat{y}^{\alpha}\, {s_*}^{\beta}\right)^{c_*} \pmod{\hat{n}} \ .$$
The pair $(\hat{x}, \hat{e})$ with $\hat{x} := \hat{y}^{\alpha}\, {s_*}^{\beta} \bmod \hat{n}$ and $\hat{e} := c_*$ is thus a solution to the flexible RSA problem.

- If $c_* = c_j$ for some $j \in \{1, \ldots, q_s\}$ (and thus $s_* = s_j$) and if $b = 1$ (i.e., Simulation 1 was run) then, remembering that $N = \hat{n}$ and $g = \hat{y}$, we get
$$\begin{cases} g^{m_j+1}\, {r_j}^E \equiv g^{m_*+1}\, {r_*}^E \pmod{N} \\ \qquad \Longrightarrow\ \hat{y}^{(m_j - m_*)} \equiv \left(\dfrac{r_*}{r_j}\right)^E \pmod{\hat{n}}\,, \\ s_j = s_* \ . \end{cases}$$
Note that we cannot have $m_* = m_j$ as otherwise we would have $r_* = r_j$ and so $(m_*, \sigma_*) = (m_j, \sigma_j)$, a contradiction. Therefore, since E is an $(\ell_m + 1)$-bit integer, we can find integers α and β by the extended Euclidean algorithm so that $\alpha\, E + \beta\, (m_j - m_*) = \gcd(E, m_j - m_*) = 1$. As a result, we have
$$\hat{y} \equiv \hat{y}^{\alpha\, E + \beta\, (m_j - m_*)} \equiv \left(\hat{y}^{\alpha}\, (r_*/r_j)^{\beta}\right)^E \pmod{\hat{n}}$$
and the pair $(\hat{x}, \hat{e})$ with $\hat{x} := \hat{y}^{\alpha}\, (r_*/r_j)^{\beta} \bmod \hat{n}$ and $\hat{e} = E$ is a solution to the flexible RSA problem.

- If $c_* = 0$ and if $b = 0$ (i.e., Simulation 0 was run) then, letting $\Lambda = \prod_i c_i$, we compute $\hat{d} := \hat{e}^{-1} \bmod \Lambda$ for an arbitrary $\hat{e} > 1$ such that $\gcd(\hat{e}, \Lambda) = 1$. So, the pair $(\hat{x}, \hat{e})$ with $\hat{x} := \hat{y}^{\hat{d}} \bmod \hat{n}$ is a solution to the flexible RSA problem:
$$\hat{x}^{\hat{e}} \equiv \hat{y}^{\hat{e}\hat{d}} \equiv \hat{y}^{\hat{e}\hat{d} \bmod \Lambda} \equiv \hat{y} \pmod{n}$$
because $c_* = 0$ implies $u = 1$ and thus $\hat{y}^{\Lambda} \bmod \hat{n} = 1$ (remember that $n = \hat{n}$ when $b = 0$).

- If $c_* = 1$ and if $b = 1$ (i.e., Simulation 1 was run) then, using extended Euclidean algorithm, we can find integers α and β s.t $\alpha\, E + \beta\, (m_* + 1) = \gcd(E, m_* +$

$1) = 1$.[‡] Hence, since $c_* = 1 = \hat{y}^{m_*+1}\, {r_*}^E \bmod N$, we get

$$\hat{y} \equiv \hat{y}^{\alpha\, E + \beta\,(m_*+1)} \equiv \left(\hat{y}^{\alpha}\, {r_*}^{-\beta}\right)^E \pmod{\hat{n}} \ .$$

Consequently, the pair $(\hat{x}, \hat{e})$ with $\hat{x} := \hat{y}^{\alpha}\, {r_*}^{-\beta} \bmod \hat{n}$ and $\hat{e} = E$ is a solution to the flexible RSA problem.

Since $\mathcal{A}$'s view is perfectly simulated, the success probability of the reduction is clearly $\epsilon_{\mathcal{A}}/2$.

For Simulation 0, we need to generate ℓ_n-bit safe RSA modulus N, (ℓ_m+1)-bit prime E, ℓ_n-bit modular inverse D and ℓ_n-bit parameter u in the key generation; we also need, for each signature query, compute r_j and s_j. We assume that we have algorithms so that the generation of safe prime is quintic, the generation of a prime is quartic and the evaluation of a modular exponentiation or of a modular inverse is cubic, in the bit-length. The evaluation of u and the $q_s\ s_j$'s amounts to $\mathrm{O}(q_s \log q_s)\ \ell_n$-bit exponentiations using the trick of [6, § 3.3]. Hence, the running time required by the reduction is (approximatively) $\tau_{\mathcal{A}} + \mathrm{O}({\ell_n}^5 + q_s \log q_s\ {\ell_n}^3)$.

For Simulation 1, further assuming that primality testing is cubic in the bit-length, we similarly obtain that the running time required by the reduction is (approximatively) $\tau_{\mathcal{A}} + \mathrm{O}({\ell_n}^5 + q_s\, {\ell_n}^4)$. $\square$

3. Encryption Schemes

A *(public-key) encryption scheme* is a set of 3 algorithms:

1. Key generation: on input a security parameter, it produces a matching pair $(\mathsf{pk}, \mathsf{sk})$ of public key and private key;

2. Encryption: on input encryption key pk, message m (and optionally random r), it outputs the ciphertext $C = \mathscr{E}(\mathsf{pk}, m\,[, r])$;

3. Decryption: on input decryption key sk and ciphertext C, it returns plaintext message $m = \mathscr{D}(\mathsf{sk}, C)$ or $\perp$ indicating that the ciphertext is invalid.

The highest security level for encryption schemes is IND-CCA2, namely *indistinguishably under adaptive chosen-ciphertext attacks*. This level of security is attained by RSA-OAEP [3]. See Fig. 4.

Indeed, it can be shown that under the RSA assumption, RSA-OAEP encryption scheme is secure in the sense of IND-CCA2 in the random oracle model.

[‡]This last case explains why $(m + 1)$ (and not merely m) appears in the description of TSS.

Key generation The public key is $\mathsf{pk} = \{N, e\}$ with $N = pq$, and the private key is $\mathsf{sk} = \{d\}$ with $d = e^{-1} \bmod \phi(N)$.

Encryption To encrypt a message m, choose a random r and compute

1. $w = (m \| 0^k) \oplus \mathcal{G}(r)$ and $t = r \oplus \mathcal{H}(w)$ [padding],
2. $C = (w \| t)^e \bmod N$ [encryption].

The ciphertext is C.

Decryption Given C, compute

1. $(w' \| t') = C^d \bmod N$,
2. $r' = \mathcal{H}(w') \oplus t'$,
3. $(m' \| z') = \mathcal{G}(r') \oplus w'$

and output $m = m'$ if $z' = 0^k$.

Figure 4. RSA-OAEP encryption scheme.

4. Conclusion

Security is defined *relatively* to a security model: the adversarial goal and the adversary's resources.

We presented several RSA-based schemes featuring provable security or, more precisely, *reductionist* security. In each case, the security was reduced to the hardness of solving the RSA problem (or some related problems), in a given security model. In particular, we detailed the TSS signature scheme whose security is solely and tightly related to the strong RSA assumption, in the standard model. In contrast to other signature schemes, it features a tight security reduction and does not require the use of specialized hash functions.

References

[1] M. Bellare. Practice-oriented provable security. In *Lectures on Data Security*, volume 1561 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag, 1999.

[2] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *1st ACM Conference on Computer and Communications Security*, pages 62–73. ACM Press, 1993.

[3] M. Bellare and P. Rogaway. Optimal asymmetric encryption – How to encrypt with RSA. In *Advances in Cryptology – EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer-Verlag, 1995.

[4] M. Bellare and P. Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In *Advances*

in Cryptology – EUROCRYPT '96, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer-Verlag, 1996.

[5] B. Chevallier-Mames and M. Joye. A practical and tightly secure signature scheme without hash function. In *Topics in Cryptology – CT-RSA 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 339–356. Springer-Verlag, 2007.

[6] J.-S. Coron, D. Lefranc, and G. Poupard. A new baby-step giant-step algorithm and some applications to cryptanalysis. In *Cryptographic Hardware and Embedded Systems – CHES 2005*, volume 3659 of *Lecture Notes in Computer Science*, pages 47–60. Springer-Verlag, 2005.

[7] R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 123–139. Springer-Verlag, 1999.

[8] N. Koblitz and A. J. Menezes. Another look at "provable security". *Journal of Cryptology*, 20(1):3–37, 2007.

[9] H. Krawczyk and T. Rabin. Chameleon signatures. In *Symposium on Network and Distributed System Security – NDSS 2000*, pages 143–154. Internet Society, 2000.

[10] P. Paillier. Impossibility proofs for RSA signatures in the standard model. In *Topics in Cryptology – CT-RSA 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 31–48. Springer-Verlag, 2007.