Universal Padding Schemes for RSA

Jean-Sébastien Coron, Marc Joye, David Naccache, and Pascal Paillier

Gemplus Card International
34 rue Guynemer, 92447 Issy-les-Moulineaux, France
{jean-sebastien.coron, marc.joye, david.naccache, pascal.paillier}@gemplus.com

Abstract. A common practice to encrypt with RSA is to first apply a padding scheme to the message and then to exponentiate the result with the public exponent; an example of this is OAEP. Similarly, the usual way of signing with RSA is to apply some padding scheme and then to exponentiate the result with the private exponent, as for example in PSS. Usually, the RSA modulus used for encrypting is different from the one used for signing. The goal of this paper is to simplify this common setting. First, we show that PSS can also be used for encryption, and gives an encryption scheme semantically secure against adaptive chosen-ciphertext attacks, in the random oracle model. As a result, PSS can be used indifferently for encryption or signature. Moreover, we show that PSS allows to safely use the same RSA key-pairs for both encryption and signature, in a concurrent manner. More generally, we show that using PSS the same set of keys can be used for both encryption and signature for any trapdoor partial-domain one-way permutation. The practical consequences of our result are important: PKIs and public-key implementations can be significantly simplified.

Key-words: Probabilistic Signature Scheme, Provable Security.

1 Introduction

A very common practice for encrypting a message m with RSA is to first apply a padding scheme μ, then raise $\mu(m)$ to the public exponent e. The ciphertext c is then:

$$c = \mu(m)^e \mod N$$

Similarly, for signing a message m, the common practice consists again in first applying a padding scheme μ' then raising $\mu'(m)$ to the private exponent d. The signature s is then:

$$s = \mu'(m)^d \mod N$$

For various reasons, it would be desirable to use the same padding scheme $\mu(m)$ for encryption and for signature: in this case, only one padding scheme needs to be implemented. Of course, the resulting padding scheme $\mu(m)$ should be provably secure for encryption and for signing. We say that a padding scheme is *universal* if it satisfies this property.

The strongest public-key encryption security notion was defined in [16] as *indistinguishability under an adaptive chosen ciphertext attack*. An adversary should not be able to distinguish between the encryption of two plaintexts, even if he can obtain the decryption of ciphertexts of his choice. For digital signature schemes, the strongest security notion was defined by Goldwasser, Micali and Rivest in [11], as *existential unforgeability under an adaptive chosen message attack*. This notion captures the property that an adversary cannot produce a valid signature, even after obtaining the signature of (polynomially many) messages of his choice.

In this paper, we show that the padding scheme PSS [4], which is originally a provably secure padding scheme for producing signatures, can also be used as a provably secure encryption scheme. More precisely, we show that PSS offers indistinguishability under an adaptive chosen ciphertext attack, in the random oracle model, under the partial-domain one-wayness of the underlying permutation. Partial-domain one-wayness, introduced in [10], is a formally stronger assumption than one-wayness. However, for RSA, partial-domain one-wayness is equivalent to (full domain) one-wayness and therefore RSA-PSS encryption is provably secure under the sole assumption that RSA is one-way.

Generally, in a given application, the RSA modulus used for encrypting is different from the RSA modulus used for signing; our setting (and real-world PKIs) would be further simplified if one could use the same set of keys for both encryption and signature (see [12]). In this paper, we show that using PSS, the same keys can be safely used for encryption and for signature.

2 Public-Key Encryption

A public-key encryption scheme is a triple of algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ where:

- $\mathcal{K}$ is a probabilistic key generation algorithm which returns random pairs of public and secret keys (pk, sk) depending on some security parameter k,

- $\mathcal{E}$ is a probabilistic encryption algorithm which takes as input a public key pk and a plaintext $M \in \mathcal{M}$, runs on a random tape $r \in \mathcal{R}$ and returns a ciphertext c. $\mathcal{M}$ and $\mathcal{R}$ stand for spaces in which messages and random strings are chosen respectively,

- $\mathcal{D}$ is a deterministic decryption algorithm which, given as input a secret key sk and a ciphertext c, returns the corresponding plaintext M, or Reject.

The strongest security notion for public-key encryption is the aforementioned notion of indistinguishability under an adaptive chosen ciphertext attack. An adversary should not be able to distinguish between the encryption of two plaintexts, even if he can obtain the decryption of ciphertexts of his choice. The attack scenario is the following:

1. The adversary $\mathcal{A}$ receives the public key pk with $(pk, sk) \leftarrow \mathcal{K}(1^\kappa)$.
2. $\mathcal{A}$ makes decryption queries for ciphertexts y of his choice.
3. $\mathcal{A}$ chooses two messages M_0 and M_1 of identical length, and receives the encryption c of M_b for a random unknown bit b.
4. $\mathcal{A}$ continues to make decryption queries. The only restriction is that the adversary cannot request the decryption of c.
5. $\mathcal{A}$ outputs a bit b', representing its "guess" on b.

The adversary's advantage is then defined as:

$$\mathsf{Adv}(\mathcal{A}) = |2 \cdot \Pr[b' = b] - 1|$$

An encryption scheme is said to be secure against adaptive chosen ciphertext attack (and denoted $\mathsf{IND\text{-}CCA2}$) if the advantage of any polynomial-time bounded adversary is a negligible function of the security parameter. Usually, schemes are proven to be

IND-CCA2 secure by exhibiting a polynomial reduction: if some adversary can break the IND-CCA2 security of the system, then the same adversary can be invoked (polynomially many times) to solve a related hard problem.

The random oracle model, introduced by Bellare and Rogaway in [2], is a theoretical framework in which any hash function is seen as an oracle which outputs a random value for each new query. Actually, a security proof in the random oracle model does not necessarily imply that a scheme is secure in the real world (see [7]). Nevertheless, it seems to be a good engineering principle to design a scheme so that it is provably secure in the random oracle model. Many encryption and signature schemes were proven to be secure in the random oracle model.

3 Encrypting with PSS-R

In this section we prove that given any trapdoor partially one-way permutation f, the encryption scheme defined by first applying PSS with message recovery (denoted PSS-R) and encrypting the result with f achieves the strongest security level for an encryption scheme, in the random oracle model.

3.1 The PSS-R Padding Scheme

PSS-R, defined in [4], is parameterized by the integers k, k_0 and k_1 and uses two hash functions:

$$H : \{0,1\}^{k-k_1} \rightarrow \{0,1\}^{k_1} \quad \text{and} \quad G : \{0,1\}^{k_1} \rightarrow \{0,1\}^{k-k_1}$$

PSS-R takes as input a $(k - k_0 - k_1)$-bit message M and a k_0-bit random integer r. As illustrated in figure 1, PSS-R outputs:

$$\mu(M,r) = \omega || s$$

where $\|$ stands for concatenation, $\omega = H(M||r)$ and $s = G(\omega) \oplus (M\|r)$. Actually, in [4], $M\|r$ is used as the argument to H and $r\|M$ is used as the mask to xor with $G(\omega)$. Here for simplicity we use $M\|r$ in both places, but the same results apply either way.

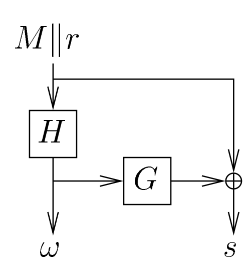

Fig. 1. The PSS-R padding scheme

3.2 The PSS-E Encryption Scheme

The new encryption scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, that we denote PSS-E, is based on μ and a k-bit trapdoor permutation f.

- $\mathcal{K}$ generates the public key f and the secret key f^{-1}.
- $\mathcal{E}(M, r)$: given a message $M \in \{0,1\}^{k-k_0-k_1}$ and a random $r \in \{0,1\}^{k_0}$, the encryption algorithm outputs the ciphertext:

$$c = f(\mu(M, r))$$

- $\mathcal{D}(c)$: the decryption algorithm recovers $(\omega, s) = f^{-1}(c)$ and then $M \| r = G(\omega) \oplus s$. If $\omega = H(M \| r)$, the algorithm returns M, otherwise it returns Reject.

3.3 The Underlying Problem

The security of PSS-E is based on the difficulty of inverting f without knowing f^{-1}. As in [10], we use two additional related problems: the partial-domain one-wayness and the set partial-domain one-wayness of f:

- (τ, ε)**-one-wayness of** f, means that for any adversary $\mathcal{A}$ who wishes to recover the full pre-image (ω, s) of $f(\omega, s)$ in time less than τ, $\mathcal{A}$'s success probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$ is upper-bounded by ε:

$$\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A}) = \Pr_{\omega,s}[\mathcal{A}(f(\omega, s)) = (\omega, s)] < \varepsilon$$

- (τ, ε)**-partial-domain one-wayness of** f, means that for any adversary $\mathcal{A}$ who wishes to recover the partial pre-image ω of $f(\omega, s)$ in time less than τ, $\mathcal{A}$'s success probability $\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A})$ is upper-bounded by ε:

$$\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A}) = \Pr_{\omega,s}[\mathcal{A}(f(\omega, s)) = \omega] < \varepsilon$$

- $(\ell, \tau, \varepsilon)$**-set partial-domain one-wayness of** f, means that for any adversary $\mathcal{A}$ who wishes to output a set of ℓ elements which contains the partial pre-image ω of $f(\omega, s)$, in time less than τ, $\mathcal{A}$'s success probability $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A})$ is upper-bounded by ε:

$$\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A}) = \Pr_{\omega,s}[\omega \in \mathcal{A}(f(\omega, s))] < \varepsilon$$

As in [10], we denote by $\mathsf{Succ}^{\mathsf{ow}}(\tau)$, (resp. $\mathsf{Succ}^{\mathsf{pd-ow}}(\tau)$ and $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\ell, \tau)$) the maximal probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$, (resp. $\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A})$ and $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A})$), over all adversaries whose running times are less than τ. For any τ and $\ell \geq 1$, we have:

$$\mathsf{Succ}^{\mathsf{s-pd-ow}}(\ell, \tau) \geq \mathsf{Succ}^{\mathsf{pd-ow}}(\tau) \geq \mathsf{Succ}^{\mathsf{ow}}(\tau)$$

Moreover, by randomly selecting any element in the set returned by the adversary against the set partial-domain one-wayness, one can break the partial-domain one-wayness with probability $1/\ell$, which gives:

$$\mathsf{Succ}^{\mathsf{pd-ow}}(\tau) \geq \mathsf{Succ}^{\mathsf{s-pd-ow}}(\ell, \tau)/\ell \tag{1}$$

We will see in Section 5 that for RSA, the three problems are polynomially equivalent.

3.4 Security of PSS-E

The following theorem shows that PSS-E is semantically secure under adaptive chosen ciphertext attacks, in the random oracle model, assuming that the underlying permutation is partially one-way.

Theorem 1. *Let $\mathcal{A}$ be a CCA2-adversary against the semantic security of PSS-E $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, with advantage ε and running time t, making q_D, q_H and q_G queries to the decryption oracle and the hash functions H and G, respectively. Then:*

$$\mathsf{Succ}^{\mathsf{pd-ow}}(t') \geq \frac{1}{q_H + q_G} \cdot \left(\varepsilon - q_H 2^{-k_0} - q_D 2^{-k_1} \right)$$

where $t' \leq t + q_H \cdot T_f$, and T_f denotes the time complexity of f.

The theorem follows from inequality (1) and the following lemma:

Lemma 1. *Using the notations introduced in theorem 1, we have:*

$$\mathsf{Succ}^{\mathsf{s-pd-ow}}(q_H + q_G, t') \geq \varepsilon - q_H \cdot 2^{-k_0} - q_D \cdot 2^{-k_1} \tag{2}$$

Proof. We describe a reduction $\mathcal{B}$ which using $\mathcal{A}$, constructs an adversary against the set partial-domain one-wayness of f. We start with a top-level description of the reduction and then show how to simulate the random oracles G, H and the decryption oracle D. Eventually we compute the success probability of $\mathcal{B}$.

Top-level description of the reduction $\mathcal{B}$:

1. $\mathcal{B}$ is given a function f and $c^* = f(\omega^*, s^*)$, for a random ω^* and s^*. $\mathcal{B}$'s goal is to output a list which contains the partial pre-image ω^* of c^*.
2. $\mathcal{B}$ runs $\mathcal{A}$ with f and gets $\{M_0, M_1\}$. It chooses a random bit b and gives c^* as a ciphertext for M_b. $\mathcal{B}$ simulates the decryption oracle H, D and G as described below.
3. $\mathcal{B}$ receives from $\mathcal{A}$ the answer b' and outputs the list of queries asked to G.

Simulation of the random oracles G, H and D.

The simulation of G and H is very simple: a random answer is returned for each new query of G and H. Moreover, when ω is the answer of a query to H, we simulate a query for ω to G, so that $G(\omega)$ is defined.

On query c to the decryption oracle, the reduction $\mathcal{B}$ looks at each query $M'||r'$ to H and computes:

$$\omega' = H(M'||r') \text{ and } s' = G(\omega') \oplus (M'\|r')$$

Then if $c = f(\omega', s')$ the reduction $\mathcal{B}$ returns M'. Otherwise, the reduction outputs Reject.

Analysis:

Since $c^* = f(\omega^*, s^*)$ is the ciphertext corresponding to M_b, we have the following constraint for the random oracles G and H:

$$H(M_b\|r^*) = \omega^* \quad \text{and} \quad G(\omega^*) = s^* \oplus (M_b\|r^*) \tag{3}$$

We denote by AskG the event: "ω^* has been asked to G" and by AskH the event: "there exists M' such that $M'||r^*$ has been queried to H".

If ω^* was never queried to G, then $G(\omega^*)$ is undefined and r^* is then a uniformly distributed random variable. Therefore the probability that there exists M' such that (M', r^*) has been asked to H is at most $q_H \cdot 2^{-k_0}$. This gives:

$$\Pr[\mathsf{AskH}|\neg\mathsf{AskG}] \le q_H \cdot 2^{-k_0} \tag{4}$$

Our simulation of D can only fail by rejecting a valid ciphertext. We denote by DBad this event. Letting $c = f(\omega, s)$ be the ciphertext queried to D and

$$M\|r = G(\omega) \oplus s$$

we reject a valid ciphertext if $H(M||r) = \omega$ while $M||r$ was never queried to H. However, if $M\|r$ was never queried to H, then $H(M\|r)$ is randomly defined. Namely if the decryption query occured before c^* was sent to the adversary, then constraint (3) does not apply and $H(M\|r)$ is randomly defined. Otherwise, if the decryption query occured after c^* was sent to the adversary, then $c \neq c^*$ implies $(M, r) \neq (M_b, r^*)$ and $H(M\|r)$ is still randomly defined. In both cases the probability that $H(M, r) = \omega$ is then 2^{-k_1}, which gives:

$$\Pr[\mathsf{DBad}] \le q_D \cdot 2^{-k_1} \tag{5}$$

Let us denote by Bad the event: "ω^* has been queried to G or (M', r^*) has been queried to H for some M' or the simulation of D has failed". Formally:

$$\mathsf{Bad} = \mathsf{AskG} \vee \mathsf{AskH} \vee \mathsf{DBad} \tag{6}$$

Let us denote by S the event: "the adversary outputs the correct value for b, *i.e.*, $b = b'$". Conditioned on $\neg\mathsf{Bad}$, our simulations of G, H and D are independent of b, and therefore $\mathcal{A}$'s view is independent of b as well. This gives:

$$\Pr[\mathsf{S}|\neg\mathsf{Bad}] = \frac{1}{2} \tag{7}$$

Moreover, conditioned on $\neg\mathsf{Bad}$, the adversary's view is the same as when interacting with (perfect) random and decryption oracles, which gives:

$$\Pr[\mathsf{S} \wedge \neg\mathsf{Bad}] \ge \frac{1}{2} + \frac{\varepsilon}{2} - \Pr[\mathsf{Bad}] \tag{8}$$

From (7) we obtain

$$\Pr[\mathsf{S} \wedge \neg\mathsf{Bad}] = \Pr[\mathsf{S}|\neg\mathsf{Bad}] \cdot \Pr[\neg\mathsf{Bad}] = \frac{1}{2}(1 - \Pr[\mathsf{Bad}])$$

which gives using (8):

$$\Pr[\mathsf{Bad}] \ge \varepsilon \tag{9}$$

From (6) we have:

$$\begin{aligned}
\Pr[\mathsf{Bad}] &\le \Pr[\mathsf{AskG} \vee \mathsf{AskH}] + \Pr[\mathsf{DBad}] \\
&\le \Pr[\mathsf{AskG}] + \Pr[\mathsf{AskH} \wedge \neg\mathsf{AskG}] + \Pr[\mathsf{DBad}] \\
&\le \Pr[\mathsf{AskG}] + \Pr[\mathsf{AskH}|\neg\mathsf{AskG}] + \Pr[\mathsf{DBad}]
\end{aligned}$$

which yields using (4), (5) and (9):

$$\Pr[\mathsf{AskG}] \ge \varepsilon - q_H \cdot 2^{-k_0} - q_D \cdot 2^{-k_1}$$

and hence (2) holds. This terminates the proof of lemma 1. $\square$

4 Signing and Encrypting with the same Public-Key

In this section we show that when using PSS, the same public key can be used for encryption and signature in a concurrent manner. For RSA, this means that the same pair (N, e) can be used for both operations. In other words, when Alice sends a message to Bob, she encrypts it using Bob's public key (N, e); Bob decrypts it using the corresponding private key (N, d). To sign a message M, Bob will use the *same* private key (N, d). As usual, anybody can verify Bob's signature using his public pair (N, e).

Although provably secure (as we will see hereafter), this is contrary to the folklore recommendation that signature and encryption keys should be distinct. This recommendation may prove useful is some cases; this is particularly true when a flaw has been found in the encryption scheme or in the signature scheme. In our case, we will prove that when using the PSS-R padding scheme, a decryption oracle does not help the attacker in forging signatures, and a signing oracle does not help the attacker in gaining information about the plaintext corresponding to a ciphertext.

Nevertheless, we advise to be very careful when implementing systems using the same keys for encrypting and signing. For example, if there are some implementation errors in a decryption server (see for example [14]), then an attacker could use this server to create forgeries.

4.1 Signature Schemes and their Security

Definition 1 (signature scheme). *A signature scheme* $(\mathtt{Gen}, \mathtt{Sign}, \mathtt{Verify})$ *is defined as follows:*

- *The key generation algorithm* $\mathtt{Gen}$ *is a probabilistic algorithm which given* 1^k*, outputs a pair of matching public and private keys,* (pk, sk)*.*

- *The signing algorithm* $\mathtt{Sign}$ *takes the message* M *to be signed, the public key* pk *and the private key* sk*, and returns a signature* $x = \mathtt{Sign}_{sk}(M)$*. The signing algorithm may be probabilistic.*

- *The verification algorithm* $\mathtt{Verify}$ *takes a message* M*, a candidate signature* x' *and* pk*. It returns a bit* $\mathtt{Verify}_{pk}(M, x')$*, equal to one if the signature is accepted, and zero otherwise. We require that if* $x \leftarrow \mathtt{Sign}_{sk}(M)$*, then* $\mathtt{Verify}_{pk}(M, x) = 1$*.*

In the *existential unforgeability under an adaptive chosen message attack* scenario, the forger can dynamically obtain signatures of messages of his choice and attempts to output a valid forgery. A *valid forgery* is a message/signature pair (M, x) such that $\mathtt{Verify}_{pk}(M, x) = 1$ whereas the signature of M was never requested by the forger.

4.2 The PSS-ES Encryption and Signature Scheme

The PSS-ES encryption and signature scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{S}, \mathcal{V})$ is based on PSS-R and a k-bit trapdoor permutation f. As for the PSS-R signature scheme, the signature scheme in PSS-ES is with message recovery: this means that the message is recovered when verifying the signature. In this case, only messages of fixed length $k - k_0 - k_1$ can be signed. To sign messages M of arbitrary length, it suffices to apply a collision-free hash function to M prior to signing.

- $\mathcal{K}$ generates the public key f and the secret key f^{-1}.

- $\mathcal{E}(M,r)$: given a message $M \in \{0,1\}^{k-k_0-k_1}$ and a random value $r \in \{0,1\}^{k_0}$, the encryption algorithm computes the ciphertext:

$$c = f(\mu(M,r))$$

- $\mathcal{D}(c)$: the encryption algorithm recovers $(\omega, s) = f^{-1}(c)$ and computes

$$M\|r = G(\omega) \oplus s$$

If $\omega = H(M\|r)$, the algorithm returns M, otherwise it returns Reject.

- $\mathcal{S}(M,r)$: given a message $M \in \{0,1\}^{k-k_0-k_1}$ and a random value $r \in \{0,1\}^{k_0}$, the signing algorithm computes the signature:

$$\sigma = f^{-1}(\mu(M,r))$$

- $\mathcal{V}(\sigma)$: given the signature σ, the verification algorithm recovers $(\omega, s) = f(\sigma)$ and computes:

$$M\|r = G(\omega) \oplus s$$

If $\omega = H(M\|r)$, the algorithm accepts the signature and returns M. Otherwise, the algorithm returns Reject.

4.3 Semantic Security

We must ensure that an adversary is still unable to distinguish between the encryption of two messages, even if he can obtain the decryption of ciphertexts of his choice, and the signature of messages of his choice. The attack scenario is consequently the same as previously, except that the adversary can also obtain the signature of messages he wants.

The following theorem, whose proof is given in Appendix A, shows that PSS-ES is semantically secure under adaptive chosen ciphertext attacks, in the random oracle model, assuming that the underlying permutation is partial domain one-way.

Theorem 2. *Let $\mathcal{A}$ be an adversary against the semantic security of PSS-ES, with success probability ε and running time t, making q_D, q_{sig}, q_H and q_G queries to the decryption oracle, the signing oracle, and the hash functions H and G, respectively. Then,* $\mathsf{Succ}^{\mathsf{pd-ow}}(t')$ *is greater than:*

$$\frac{1}{q_H + q_G + q_{sig}} \left(\varepsilon - (q_H + q_{sig}) \cdot 2^{-k_0} - q_D 2^{-k_1} - (q_H + q_{sig})^2 \cdot 2^{-k_1} \right)$$

where $t' \leq t + (q_H + q_{sig}) \cdot T_f$, and T_f denotes the time complexity of f.

4.4 Unforgeability

For signature schemes, the strongest security notion is the previously introduced existential unforgeability under an adaptive chosen message attack. An attacker cannot produce a valid signature, even after obtaining the signature of (polynomially many) messages of his choice. Here the adversary can also also obtain the decryption of ciphertexts of his choice under the same public-key. Consequently, the attack scenario is the following:

1. The adversary $\mathcal{A}$ receives the public key pk with $(pk, sk) \leftarrow \mathcal{K}(1^\kappa)$.
2. $\mathcal{A}$ makes signature queries for messages M of his choice. Additionally, he makes decryption queries for ciphertexts y of his choice.
3. $\mathcal{A}$ outputs the signature of a message M' which was not queried for signature before.

An encryption-signature scheme is said to be secure against chosen-message attacks if for any polynomial-time bounded adversary, the probability to output a forgery is negligible.

The following theorem shows that PSS-ES is secure against an adaptive chosen message attack. The proof is similar to the security proof of PSS [4] and is given in Appendix B.

Theorem 3. *Let $\mathcal{A}$ be an adversary against the unforgeability of PSS-ES, with success probability ε and running time t, making q_D, q_{sig}, q_H and q_G queries to the decryption oracle, the signing oracle, and the hash oracles H and G, respectively. Then $\mathsf{Succ}^{\mathsf{ow}}(t')$ is greater than:*

$$\frac{1}{q_H}\left(\varepsilon - ((q_H + q_{sig})^2 + q_D + 1) \cdot 2^{-k_1}\right) \tag{10}$$

where $t' \leq t + (q_H + q_{sig}) \cdot T_f$, and T_f denotes the time complexity of f.

5 Application to RSA

5.1 The RSA cryptosystem

The RSA cryptosystem, invented by Rivest, Shamir and Adleman [17], is the most widely used cryptosystem today. In this section, we show that by virtue of RSA's homomorphic properties, the partial-domain one-wayness of RSA is equivalent to the one-wayness of RSA. This enables to prove that the encryption scheme RSA-PSS-E and the encryption and signature scheme RSA-PSS-ES are semantically secure against chosen ciphertext attacks, in the random oracle model, assuming that inverting RSA is hard.

Definition 2 (The RSA Primitive). *The RSA primitive is a family of trapdoor permutations, specified by:*

- The RSA generator $\mathcal{RSA}$, which on input 1^k, randomly selects two distinct $k/2$-bit primes p and q and computes the modulus $N = p \cdot q$. It randomly picks an encryption exponent $e \in \mathbb{Z}^*_{\phi(N)}$, computes the corresponding decryption exponent $d = e^{-1} \bmod \phi(N)$ and returns (N, e, d).

- The encryption function $f : \mathbb{Z}^*_N \to \mathbb{Z}^*_N$ defined by $f(x) = x^e \mod N$.

- The decryption function $f^{-1} : \mathbb{Z}^*_N \to \mathbb{Z}^*_N$ defined by $f^{-1}(y) = y^d \mod N$.

In the following, we state our result in terms of the RSA primitive with a randomly chosen public exponent. The same results apply to the common practice of choosing a small public exponent. Actually, using Coppersmith's algorithm [8] as in [18] for OAEP [3], it would be possible to obtain tigther bounds for a small public exponent.

5.2 Partial-domain One-Wayness of RSA

The following lemma shows that the partial-domain one-wayness of RSA is equivalent to the one-wayness of RSA. This is a generalization of the result that appears in [10] for OAEP and in [5] for SAEP$^+$, wherein the size of the partial pre-image is always greater than half the size of the modulus. [10] relies upon lattice reduction techniques for lattices of dimension 2. Here the partial pre-image can be smaller than half the size of the modulus (*e.g* a 160-bit pre-image for a 1024-bit modulus), so we must consider lattices of higher dimension. The extension was announced in [10] and [5], even if the proper estimates were not worked out.

The technique goes as follows. Given $y = x^e \mod N$, we must find x. We obtain the most significant bits of $x \cdot \alpha_i \mod N$ for random integers $\alpha_i \in \mathbb{Z}_N$, by querying for the partial pre-image of $y_i = y \cdot (\alpha_i)^e \mod N$. Finding x from the most significant bits of the $x \cdot \alpha_i \mod N$ is a Hidden Number Problem modulo N. We use an algorithm similar to [6] to efficiently recover x.

Lemma 2. *Let $\mathcal{A}$ be an algorithm that on input y, outputs a q-set containing the k_1 most significant bits of $y^d \mod N$, within time bound t, with probability ε, where $2^{k-1} \le N < 2^k$, $k_1 \ge 64$ and $k/(k_1)^2 \le 2^{-6}$. Then there exists an algorithm $\mathcal{B}$ that solves the RSA problem with success probability ε' within time bound t', where:*

$$\varepsilon' \ge \varepsilon \cdot (\varepsilon^{n-1} - 2^{-k/8}) \tag{11}$$

$$t' \le n \cdot t + q^n \cdot \texttt{poly}(k)$$

$$n = \left\lceil \frac{5k}{4k_1} \right\rceil$$

Proof. Algorithm $\mathcal{B}$ receives y as input and must output $y^d \mod N$. It generates the integers $\alpha_i \in \mathbb{Z}_N$ at random for $1 \le i \le n-1$, where n is an integer which will be determined later.

Let $y_0 = y$ and $y_i = y \cdot (\alpha_i)^e \mod N$ for $1 \le i \le n-1$. We write for $0 \le i \le n-1$:

$$(y_i)^d = \omega_i \cdot 2^{k-k_1} + s_i \mod N$$

where $0 \le s_i < 2^{k-k_1}$. Letting $k_2 = k - k_1$, we obtain for $1 \le i \le n-1$:

$$\alpha_i \cdot (\omega_0 \cdot 2^{k_2} + s_0) = \omega_i \cdot 2^{k_2} + s_i \mod N$$

Therefore letting $c_i = 2^{k_2} \cdot (\alpha_i \cdot \omega_0 - \omega_i) \mod N$, we obtain the following system of $n-1$ equations in the n unknown s_i:

$$\mathcal{S}: \quad s_i - \alpha_i \cdot s_0 = c_i \mod N \quad \text{for } 1 \le i \le n-1 \tag{12}$$

The following lemma, whose proof is given in appendix C, shows that given the c_i and α_i, the s_i can be recovered in time polynomial in k. We denote by:

$$\|\boldsymbol{x}\|_\infty = \max |x_i|$$

the infinite norm of vector $\boldsymbol{x}$.

Lemma 3. *If the previous set $\mathcal{S}$ of equations has a solution $\boldsymbol{s} = (s_0, \ldots, s_{n-1})$ such that $\|\boldsymbol{s}\|_\infty < 2^{k_2}$, then for all values of $\boldsymbol{\alpha}$, except a fraction:*

$$\frac{2^{n\cdot(k_2+n+2)}}{N^{n-1}} \tag{13}$$

of them, this solution is unique and can be computed in time polynomial in n and in the size of N.

Consequently, algorithm $\mathcal{B}$ runs n times algorithm $\mathcal{A}$ with input y_i. It obtains n sets of q integers, each set containing ω_i. Then it applies q^n times the algorithm of lemma 3, one execution of the algorithm enabling to recover the s_i, with probability:

$$\varepsilon' \geq \varepsilon \cdot \left(\varepsilon^{n-1} - \frac{2^{n\cdot(k_2+n+2)}}{N^{n-1}} \right)$$

In appendix D we show that taking $n = \lceil 5k/(4k_1) \rceil$ we obtain:

$$\frac{2^{n\cdot(k_2+n+2)}}{N^{n-1}} \leq 2^{-k/8} \tag{14}$$

which gives (11). $\square$

5.3 RSA-PSS-E and RSA-PSS-ES

The RSA-PSS-E encryption scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ based on the PSS-R padding μ with parameters k, k_0, and k_1 is defined as follows:

- $\mathcal{K}$ generates a $k+1$-bit RSA modulus and exponents e and d. The public key is (N, e) and the private key is (N, d).

- $\mathcal{E}(M, r)$: given a message $M \in \{0,1\}^{k-k_0-k_1}$ and a random $r \in \{0,1\}^{k_0}$, the encryption algorithm outputs the ciphertext:

$$c = (\mu(M, r))^e \mod N$$

- $\mathcal{D}(c)$: the decryption algorithm recovers $x = c^d \mod N$. It returns Reject if the most significant bit of x is not zero. It writes x as $0\|\omega\|s$ where ω is a k_1-bit string and s is a $k - k_1$ bit string. It writes $M\|r = G(\omega) \oplus s$. If $\omega = H(M\|r)$, the algorithm returns M, otherwise it returns Reject.

The RSA-PSS-ES encryption and signature scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{S}, \mathcal{V})$ is defined as follows:

- $\mathcal{K}$, $\mathcal{E}(M, r)$ and $\mathcal{D}(c)$ are identical to RSA-PSS-E.

- $\mathcal{S}(M, r)$: given a message $M \in \{0,1\}^{k-k_0-k_1}$ and a random value $r \in \{0,1\}^{k_0}$, the signing algorithm computes the signature:

$$\sigma = \mu(M, r)^d \mod N$$

- $\mathcal{V}(\sigma)$: given the signature σ, the verification algorithm recovers $x = \sigma^e \mod N$. It returns Reject if the most significant bit of x is not zero. It writes x as $0\|\omega\|s$ where ω is a k_1-bit string and s is a $k - k_1$ bit string. It writes $M\|r = G(\omega) \oplus s$. If $\omega = H(M\|r)$, the algorithm accepts the signature and returns M, otherwise it returns Reject.

5.4 Security of RSA-PSS-E and RSA-PSS-ES

Combining lemma 1 and lemma 2, we obtain the following theorem which shows that the encryption scheme RSA-PSS-E is provably secure in the random oracle model, assuming that inverting RSA is hard.

Theorem 4. *Let $\mathcal{A}$ be a CCA2-adversary against the semantic security of the RSA-PSS-E scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, with advantage ε and running time t, making q_D, q_H and q_G queries to the decryption oracle and the hash function H and G, respectively. We assume that $k_1 \geq 64$ and $k/(k_1)^2 \leq 2^{-6}$. Then we can invert RSA with probability ε' greater than:*

$$\varepsilon' \geq \left(\varepsilon - q_H \cdot 2^{-k_0} - q_D 2^{-k_1}\right)^n - 2^{-k/8}$$

within time bound $t' \leq n \cdot t + (q_H + q_G)^n \cdot \mathtt{poly}(k) + n \cdot q_H \cdot \mathcal{O}(k^3)$, where $n = \lceil 5k/(4k_1) \rceil$.

We obtain a similar theorem for the semantic security of the RSA-PSS-ES encryption and signature scheme (from Lemma 2 and Lemma 4 in appendix A).

Theorem 5. *Let $\mathcal{A}$ be a CCA2-adversary against the semantic security of the RSA-PSS-ES scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{S}, \mathcal{V})$, with advantage ε and running time t, making q_D, q_{sig}, q_H and q_G queries to the decryption oracle, the signing oracle and the hash function H and G, respectively. Provided that $k_1 \geq 64$ and $k/(k_1)^2 \leq 2^{-6}$, RSA can be inverted with probability ε' greater than:*

$$\varepsilon' \geq \left(\varepsilon - (q_H + q_{sig}) \cdot 2^{-k_0} - (q_D + (q_H + q_{sig})^2) \cdot 2^{-k_1}\right)^n - 2^{-k/8}$$

within time bound $t' \leq n \cdot t + (q_H + q_G + q_{sig})^n \cdot \mathtt{poly}(k)$, where $n = \lceil 5k/(4k_1) \rceil$.

For the unforgeability of the RSA-PSS-ES encryption and signature scheme, we obtain a better security bound than the general result of Theorem 3, by relying upon the homomorphic properties of RSA. The proof of the following theorem is similar to the security proof of PSS in [4] and is given in appendix E.

Theorem 6. *Let $\mathcal{A}$ be an adversary against the unforgeability of the PSS-ES scheme $(\mathcal{K}, \mathcal{E}, \mathcal{D}, \mathcal{S}, \mathcal{V})$, with success probability ε and running time t, making q_D, q_{sig}, q_H and q_G queries to the decryption oracle, the signing oracle, and the hash functions H and G, respectively. Then RSA can be inverted with probability ε' greater than:*

$$\varepsilon' \geq \varepsilon - \left((q_H + q_{sig})^2 + q_D + 1\right) \cdot (2^{-k_0} + 2^{-k_1}) \tag{15}$$

within time bound $t' \leq t + (q_H + q_{sig}) \cdot \mathcal{O}(k^3)$.

Note that as for OAEP [10], the security proof for encrypting with PSS is far from being tight. This means that it does not provide a meaningful security result for a moderate size modulus (*e.g.*, 1024 bits). For the security proof to be meaningful in practice, we recommend to take $k_1 \geq k/2$ and to use a larger modulus (*e.g.*, 2048 bits).

6 Conclusion

In all existing PKIs different padding formats are used for encrypting and signing; moreover, it is recommended to use different keys for encrypting and signing. In this paper we have proved that the PSS padding scheme used in PKCS#1 v.2.1 [15] and IEEE P1363 [13] can be safely used for encryption as well. We have also proved that the same key pair can be safely used for both signature and encryption. The practical consequences of this are significant: besides halving the number of keys in security systems and simplifying their architecture, our observation allows resource-constrained devices such as smart cards to use the same code for implementing both signature and encryption.

Acknowledgements

We wish to thank Jacques Stern for pointing out an error in an earlier version of this paper, and the anonymous referees of Crypto 2002 for their useful comments.

References

1. L. Babai, *On Lovász' lattice reduction and the nearest lattice point problem*, Combinatorica 6 (1986), 1-14.
2. M. Bellare and P. Rogaway, *Random oracles are practical: a paradigm for designing efficient protocols*. Proceedings of the First Annual Conference on Computer and Commmunications Security, ACM, 1993.
3. M. Bellare and P. Rogaway, *Optimal Asymmetric Encryption*, Proceedings of Eurocrypt'94, LNCS vol. 950, Springer-Verlag, 1994, pp. 92–111.
4. M. Bellare and P. Rogaway, *The exact security of digital signatures - How to sign with RSA and Rabin*. Proceedings of Eurocrypt'96, LNCS vol. 1070, Springer-Verlag, 1996, pp. 399-416.
5. D. Boneh, *Simplified OAEP for the RSA and Rabin functions*, Prooceedings of Crypto 2001, LNCS vol 2139, pp. 275-291, 2001.
6. D. Boneh and R. Venkatesan, *Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes*. Proceedings of Crypto '96, pp. 129-142, 1996.
7. R. Canetti, O. Goldreich and S. Halevi, *The random oracle methodology, revisited*, STOC' 98, ACM, 1998.
8. D. Coppersmith, *Finding a small root of a univariate modular equation*, in Eurocrypt '96, LNCS 1070.
9. J.S. Coron, M. Joye, D. Naccache and P. Paillier, *Universal padding schemes for RSA*. Full version of this paper. Cryptology ePrint Archive, `http://eprint. iacr.org`.
10. E. Fujisaki, T. Okamoto, D. Pointcheval and J. Stern, *RSA-OAEP is secure under the RSA assumption*, Proceedings of Crypto' 2001, LNCS vol. 2139, Springer-Verlag, 2001, pp. 260-274.
11. S. Goldwasser, S. Micali and R. Rivest, *A digital signature scheme secure against adaptive chosen-message attacks*, SIAM Journal of computing, 17(2), pp. 281-308, April 1988.
12. S. Haber and B. Pinkas, *Combining Public Key Cryptosystems*, Proceedings of the ACM Computer and Security Conference , November 2001.
13. IEEE P1363a, *Standard Specifications For Public Key Cryptography: Additional Techniques*, available at `http://www.manta.ieee.org/groups/1363`
14. J. Manger, *A chosen ciphertext attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as Standardized in PKCS #1 v2.0*. Proceedings of Crypto 2001, LNCS 2139, pp. 230-238, 2001.
15. PKCS #1 v2.1, *RSA Cryptography Standard (draft)*, available at `http: www.rsasecurity.com /rsalabs/pkcs`.
16. C.Rackoff and D. Simon, *Noninteractive zero-knowledge proof of knowledge and chosen ciphertext attack*. Advances in Cryptology, Crypto '91, pages 433-444, 1991.
17. R. Rivest, A. Shamir and L. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, CACM 21, 1978.
18. V. Shoup, *OAEP reconsidered*, Proceedings of Crypto 2001, LNCS vol. 2139, pp 239-259, 2001.

A Proof of Theorem 2

The theorem follows from inequality (1) and the following lemma.

Lemma 4. *Let $\mathcal{A}$ be an adversary against the semantic security of PSS-ES, with success probability ε and running time t, making q_D, q_{sig}, q_H and q_G queries to the decryption oracle, the signing oracle, and the hash functions H and G, respectively. Then, the success probability $\mathsf{Succ}^{\mathsf{s-pd-ow}}(q_G, t')$ is greater than:*

$$\varepsilon - (q_H + q_{sig}) \cdot 2^{-k_0} - q_D 2^{-k_1} - (q_H + q_{sig})^2 \cdot 2^{-k_1}$$

where $t' \leq t + (q_H + q_{sig}) \cdot T_f$, and T_f denotes the time complexity of f.

Proof. The proof is very similar to the proof of lemma 1. The top-level description of the reduction $\mathcal{B}$ is the same and the simulation of the decryption oracle is the same. However, oracles H and G are simulated differently. Instead of simulating H and G so that $\mu(M, r) = y$ is a random integer, we simulate H and G so that $\mu(M, r) = f(x)$ for a known random x, which allows to answer the signature query for M.

Simulation of oracles G and H and signing oracle:

When receiving the query $M \| r$ to H, we generate a random $x \in \{0,1\}^k$ and compute $y = f(x)$. We denote $y = \omega \| s$. If ω never appeared before, we let $G(\omega) = s \oplus (M \| r)$ and return ω, otherwise we abort.

When receiving a query ω for G, if $G(\omega)$ has already been defined, we return $G(\omega)$, otherwise we return a random $(k - k_1)$-bit integer.

When we receive a signature query for M, we generate a random k_0-bit integer r. If $M \| r$ was queried to H before, we know ω, s, y and x such that:

$$H(M \| r) = \omega \quad \text{and} \quad G(\omega) = s \oplus (M \| r) \quad \text{and} \quad y = f(x) = \omega \| s$$

so we return the corresponding signature x. If $M \| r$ was never queried before, we simulate an H-query for $M \| r$ as previously: we pick a random $x \in \{0,1\}^k$ and compute $y = f(x)$. We denote $y = \omega \| s$. If ω never appeared before, we let $H(M \| r) = \omega$, $G(\omega) = s \oplus (M \| r)$ and return the signature x, otherwise we abort.

Analysis

As in lemma 1, we denote by AskG the event: "ω^* has been asked to G" and by AskH the event: "there exists M' such that $M' || r^*$ has been queried to H"; we denote by DBad the event: "a valid ciphertext has been rejected by our simulation of the decryption oracle D". Moreover, we denote by SBad the event: "the reduction aborts when answering a H-oracle query or a signature query". As previously, we have:

$$\Pr[\mathsf{AskH} | \neg \mathsf{AskG}] \leq (q_H + q_{sig}) \cdot 2^{-k_0}$$

and

$$\Pr[\mathsf{DBad}] \leq q_D \cdot 2^{-k_1}$$

When answering an H-oracle query or a signature query, the integer ω which is generated is uniformly distributed because f is a permutation. Moreover, at most $q_H + q_{sig}$

values of ω can appear during the reduction. Therefore the probability that the reduction aborts when answering an H-oracle query or a signature query is at most $(q_H + q_{sig}) \cdot 2^{-k_1}$, which gives:

$$\Pr[\mathsf{SBad}] \leq (q_H + q_{sig})^2 \cdot 2^{-k_1}$$

We denote by Bad the event:

$$\mathsf{Bad} = \mathsf{AskG} \vee \mathsf{AskH} \vee \mathsf{DBad} \vee \mathsf{SBad}$$

Let S denote the event: "the adversary outputs the correct value for b, *i.e.* $b = b'$". Conditioned on $\neg\mathsf{Bad}$, our simulation of oracles G, H, D and of the signing oracle are independent of b, and therefore the adversary's view is independent of b. This gives:

$$\Pr[\mathsf{S}|\neg\mathsf{Bad}] = \frac{1}{2} \tag{16}$$

Moreover, conditioned on $\neg\mathsf{Bad}$, the adversary's view is the same as when interacting with (perfect) random oracles, decryption oracle and signing oracle, which gives:

$$\Pr[\mathsf{S} \wedge \neg\mathsf{Bad}] \geq \frac{1}{2} + \frac{\varepsilon}{2} - \Pr[\mathsf{Bad}] \tag{17}$$

which yields as in Lemma 1:

$$\Pr[\mathsf{Bad}] \geq \varepsilon \tag{18}$$

and eventually:

$$\Pr[\mathsf{AskG}] \geq \varepsilon - (q_H + q_{sig}) \cdot 2^{-k_0} - q_D \cdot 2^{-k_1} - (q_H + q_{sig})^2 \cdot 2^{-k_1}$$

B Proof of Theorem 3

From $\mathcal{A}$ we construct an algorithm $\mathcal{B}$, which receives as input c and outputs η such that $c = f(\eta)$.

Top-level description of the reduction $\mathcal{B}$:

1. $\mathcal{B}$ is given a function f and $c = f(\eta)$, for a random integer η.
2. $\mathcal{B}$ selects uniformly at random an integer $j \in [1, q_H]$.
3. $\mathcal{B}$ runs $\mathcal{A}$ with f. It simulates the decryption oracle, the signing oracle and random oracles H and G as described below. $\mathcal{B}$ maintains a counter i for the i-th query $M_i\|r_i$ to H. The oracles H and G are simulated in such a way that if $i = j$ then $\mu(M_i\|r_i) = c$.
4. $\mathcal{B}$ receives from $\mathcal{A}$ a forgery σ. Letting M and r be the corresponding message and random, if $(M, r) = (M_j, r_j)$ then $f(\sigma) = \mu(M_j\|r_j) = c$ and $\mathcal{B}$ outputs σ.

Simulation of the oracles G, H, D and signing oracle:

When receiving the i-th query $M_i\|r_i$ to H, we distinguish two cases: if $i \neq j$, we generate a random $x_i \in \{0,1\}^k$ and compute $y_i = f(x_i)$. If $i = j$, we let $y_i = c$. In both cases we denote $y_i = \omega_i\|s_i$. If ω_i never appeared before, we let $G(\omega_i) = s_i \oplus (M_i\|r_i)$ and return ω_i, otherwise we abort.

When receiving a query ω for G, if $G(\omega)$ has already been defined, we return $G(\omega)$, otherwise we return a random $(k - k_1)$-bit integer.

When we receive a signature query for M, we generate a random k_0-bit integer r. If $M\|r$ was queried to H before, we have $M\|r = M_i\|r_i$ for some i. If $i \neq j$, we have:

$$H(M_i\|r_i) = \omega_i, \quad G(\omega_i) = s_i \oplus (M_i\|r_i) \text{ and } y_i = \omega_i\|s_i = f(x_i)$$

so we return the corresponding signature x_i, otherwise we abort. If $M\|r$ was never queried before, we simulate an H-query for $M\|r$ as previously: we generate a random $x \in \{0,1\}^k$ and compute $y = f(x)$. We denote $y = \omega\|s$. If ω never appeared before, we let $H(M\|r) = \omega$ and $G(\omega) = s \oplus (M\|r)$ and return the signature x, otherwise we abort.

The simulation of the decryption oracle is identical to that of Lemma 1.

Analysis:

Let σ be the forgery sent by the adversary. If ω was not queried to G, we simulate a query to G as previously. Let $\omega\|s = f(\sigma)$ and $M\|r = G(\omega) \oplus s$. If $M\|r$ was never queried to H, then $H(M\|r)$ is undefined because there was no signature query for M; the probability that $H(M\|r) = \omega$ is then 2^{-k_1}. Otherwise, let $(M, r) = (M_i, r_i)$ be the corresponding query to H. If $i = j$, then $\mu(M_j, r_j) = c = f(\sigma)$ and $\mathcal{B}$ succeeds in inverting f.

Conditioned on $i = j$, our simulation of H and the signing oracle are perfect, unless some ω appears twice, which happens with probability less than $(q_H + q_{sig})^2 \cdot 2^{-k_1}$. As in lemma 1, our simulation of D fails with probability less than $q_D \cdot 2^{-k_1}$. Consequently, the reduction $\mathcal{B}$ succeeds with probability greater than:

$$\frac{1}{q_H} \cdot \left(\varepsilon - 2^{-k_1} - (q_H + q_{sig})^2 \cdot 2^{-k_1} - q_D \cdot 2^{-k_1}\right)$$

which gives (10).

C Proof of Lemma 3

Let $\boldsymbol{b_1}, \ldots, \boldsymbol{b_d} \in \mathbb{Z}^n$ be linearly independent vectors. A *lattice* L spanned by the vectors $(\boldsymbol{b_1}, \ldots, \boldsymbol{b_d})$ is the set of all integer linear combinations of $\boldsymbol{b_1}, \ldots, \boldsymbol{b_d}$. The integer d is called the *rank* of the lattice. We say that the lattice is of full rank if $n = d$. We denote by $\|L\|_\infty$ the infinite norm of the shortest non-zero vector of L.

Given $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{n-1}) \in (\mathbb{Z}_N)^{n-1}$, consider the set:

$$L(\boldsymbol{\alpha}) = \{\boldsymbol{s} = (s_0, \ldots, s_{n-1}) \in \mathbb{Z}^n | s_i - \alpha_i \cdot s_0 = 0 \mod N \text{ for all } 1 \leq i \leq n-1\}$$

The set L is a full rank lattice spanned by the n vectors:

$$(1, \alpha_1, \ldots, \alpha_{n-1}), (0, N, 0, \ldots, 0), \cdots, (0, \ldots, 0, N) \tag{19}$$

The proof of lemma 3 is based on the following three lemmas:

Lemma 5. *The probability over* $\boldsymbol{\alpha} \in \mathbb{Z}_N^{n-1}$ *that* $\|L(\boldsymbol{\alpha})\|_\infty < C$ *is less than*

$$\frac{(3C)^n}{N^{n-1}}$$

Proof. To each $L(\boldsymbol{\alpha})$ such that $\|L(\boldsymbol{\alpha})\|_\infty < C$ we can associate a shortest vector $\boldsymbol{b}(\boldsymbol{\alpha})$ such that $\|\boldsymbol{b}(\boldsymbol{\alpha})\|_\infty < C$. There are at most $(2C+1)^n$ such vectors.

Let b_0 be the first component of $\boldsymbol{b(\alpha)}$. If $b_0 = 0 \mod N$, then all the components of $\boldsymbol{b(\alpha)}$ are equal to 0 modulo N. This gives $\|\boldsymbol{b(\alpha)}\|_\infty \geq N$, and the first vector in (19) is shorter for $\|\|_\infty$ than $\boldsymbol{b(\alpha)}$. Therefore $b_0 \neq 0 \mod N$.

If b_0 is invertible modulo N, this uniquely determines $\boldsymbol{\alpha}$. Otherwise, let p and q be the prime factors of N. If $b_0 = 0 \mod p$, then $b_0 \neq 0 \mod q$ and this uniquely determines $\boldsymbol{\alpha}$ modulo q, so there are at most p^{n-1} possible values for $\boldsymbol{\alpha}$. Moreover, all the components of $\boldsymbol{b(\alpha)}$ are equal to 0 modulo p, and for any C the number of such vectors $\boldsymbol{b(\alpha)}$ is at most:

$$\left(2 \cdot \left\lfloor \frac{C}{p} \right\rfloor + 1\right)^n - 1 \leq \left(\frac{3C}{p}\right)^n$$

which corresponds to at most:

$$\left(\frac{3C}{p}\right)^n \cdot p^{n-1} = \frac{(3C)^n}{p}$$

possible values for $\boldsymbol{\alpha}$. The same holds if $b_0 = 0 \mod q$. Therefore there are at most:

$$(2C+1)^n + (3C)^n \cdot \left(\frac{1}{p} + \frac{1}{q}\right) \leq (3C)^n$$

vectors $\boldsymbol{\alpha}$ such that $\|L(\boldsymbol{\alpha})\|_\infty < C$. □

Lemma 6. *If $\|L(\boldsymbol{\alpha})\|_\infty \geq 2 \cdot B$, then the solution $\boldsymbol{s}$ of the system $\mathcal{S}$ with $\|\boldsymbol{s}\|_\infty < B$ is unique and is equal to $\boldsymbol{T} - \boldsymbol{P}$, where $\boldsymbol{T} = (0, c_1, \ldots, c_{n-1})$ and $\boldsymbol{P}$ is the closest vector to $\boldsymbol{T}$ for $\|\|_\infty$.*

Proof. Let $\boldsymbol{s'}$ be another solution of $\mathcal{S}$ with $\|\boldsymbol{s'}\|_\infty < B$. Then $\boldsymbol{s} - \boldsymbol{s'} \in L(\boldsymbol{\alpha})$ and $\|\boldsymbol{s} - \boldsymbol{s'}\|_\infty < 2 \cdot B$ which gives $\boldsymbol{s} = \boldsymbol{s'}$ since $\|L(\boldsymbol{\alpha})\|_\infty \geq 2 \cdot B$

Let $\boldsymbol{s'} = \boldsymbol{T} - \boldsymbol{P}$ where $\boldsymbol{P} \in L(\boldsymbol{\alpha})$ is a closest vector to $\boldsymbol{T}$ for $\|\|_\infty$. Since $\boldsymbol{T} - \boldsymbol{s}$ is a vector of $L(\boldsymbol{\alpha})$, we have:

$$\|\boldsymbol{s'}\|_\infty = \|\boldsymbol{T} - \boldsymbol{P}\|_\infty \leq \|\boldsymbol{T} - (\boldsymbol{T} - \boldsymbol{s})\|_\infty = \|\boldsymbol{s}\|_\infty$$

$$\|\boldsymbol{s'} - \boldsymbol{s}\|_\infty \leq \|\boldsymbol{s'}\|_\infty + \|\boldsymbol{s}\|_\infty \leq 2\|\boldsymbol{s}\|_\infty < 2 \cdot B$$

and so $\boldsymbol{s'} = \boldsymbol{s}$. □

Lemma 7. *Let $(\boldsymbol{b_1}, \ldots, \boldsymbol{b_n})$ be a basis of a lattice $L \subset \mathbb{Z}^n$ such that $\|L\|_\infty \geq B \cdot (1 + \sqrt{n} \cdot 2^{n/2})$ and $\boldsymbol{T}$ a vector which distance to L for $\|\|_\infty$ is strictly less than B. There exists a polynomial-time algorithm taking as input $(\boldsymbol{b_1}, \ldots, \boldsymbol{b_n})$ and $\boldsymbol{T}$ and outputting a closest vector $\boldsymbol{P} \in L$ to $\boldsymbol{T}$ for $\|\|_\infty$.*

Proof. The proof is based on the following theorem:

Theorem 7 (Babai [1]). *There exists a polynomial time algorithm which, given a basis* $(\boldsymbol{b_1}, \dots, \boldsymbol{b_n})$ *of a lattice* $L \subset \mathbb{Z}^n$*, approximates the closest vector problem for the Euclidean norm to a factor* $2^{n/2}$*.*

Let $\boldsymbol{P'}$ be the vector obtained by running Babai's algorithm on $(\boldsymbol{b_1}, \dots, \boldsymbol{b_n})$ and $\boldsymbol{T}$. Let $\boldsymbol{P}$ a closest vector to $\boldsymbol{T}$ for $\|\|_\infty$. We show that $\boldsymbol{P'} = \boldsymbol{P}$.

Letting $\boldsymbol{P''}$ be a closest vector to $\boldsymbol{T}$ for the Euclidean norm, we have:

$$\|\boldsymbol{T} - \boldsymbol{P'}\| \le 2^{n/2} \|\boldsymbol{T} - \boldsymbol{P''}\|$$

Moreover, since $\boldsymbol{P''}$ is a closest vector to $\boldsymbol{T}$ for $\|\|$, we have:

$$\|\boldsymbol{T} - \boldsymbol{P''}\| \le \|\boldsymbol{T} - \boldsymbol{P}\|$$

The distance of $\boldsymbol{T}$ to L for $\|\|_\infty$ is strictly less than B, therefore:

$$\|\boldsymbol{T} - \boldsymbol{P}\|_\infty < B$$

This gives:

$$\|\boldsymbol{T} - \boldsymbol{P'}\|_\infty \le \|\boldsymbol{T} - \boldsymbol{P'}\| \le 2^{n/2}\|\boldsymbol{T} - \boldsymbol{P}\| \le \sqrt{n} \cdot 2^{n/2} \|\boldsymbol{T} - \boldsymbol{P}\|_\infty < B\sqrt{n} \cdot 2^{n/2}$$

and eventually

$$\|\boldsymbol{P} - \boldsymbol{P'}\|_\infty \le \|\boldsymbol{P} - \boldsymbol{T}\|_\infty + \|\boldsymbol{T} - \boldsymbol{P'}\|_\infty < B(1 + \sqrt{n} \cdot 2^{n/2})$$

and so $\boldsymbol{P} = \boldsymbol{P'}$. □

Resuming the proof of lemma 3, we take $B = 2^{k_2}$ and $C = 2^{k_2}(1 + \sqrt{n} \cdot 2^{n/2})$. We consider the lattices $L(\boldsymbol{\alpha})$ such that $\|L(\boldsymbol{\alpha})\|_\infty \ge C$. From lemma 5, and using:

$$3C \le 2^{k_2+n+2}$$

the proportion of lattices $L(\boldsymbol{\alpha})$ such that $\|L(\boldsymbol{\alpha})\|_\infty < C$ is smaller than:

$$\frac{2^{n\cdot(k_2+n+2)}}{N^{n-1}}$$

From lemma 6 the solution $\boldsymbol{s}$ of the system $\mathcal{S}$ with $\|\boldsymbol{s}\|_\infty < 2^{k_2}$ is unique and equal to $\boldsymbol{T} - \boldsymbol{P}$, where $\boldsymbol{T} = (0, c_1, \dots, c_{n-1})$ and $\boldsymbol{P}$ is the closest vector to $\boldsymbol{T}$ for $\|\|_\infty$. From lemma 7, and using the basis (19) for $L(\boldsymbol{\alpha})$, we can compute $\boldsymbol{P}$ in time polynomial in n and in the size of N.

D Proof of Inequality (14)

We assume that:

$$k_1 \ge 64 \quad \text{and} \quad k \le 2^{-6} \cdot (k_1)^2 \tag{20}$$

We have:

$$\begin{aligned}\frac{2^{n\cdot(k_2+n+2)}}{N^{n-1}} &\le 2^{n\cdot(k-k_1+n+2)-(n-1)\cdot(k-1)} \\ &\le 2^{n\cdot(-k_1+n+3)+k-1}\end{aligned}$$

Letting $f(x) = x \cdot (-k_1 + x + 3) + k - 1$, we have $f'(x) = -k_1 + 2 \cdot x + 3$. For $0 \leq x \leq 5k/(4k_1) + 1$ and using (20), we obtain $f'(x) \leq 0$. We take:

$$n = \left\lceil \frac{5k}{4k_1} \right\rceil$$

f is then a decreasing function for $0 \leq x \leq n$, therefore:

$$f\left(\frac{5k}{4k_1}\right) \geq f(n)$$

which yields using (20):

$$f(n) \leq -k/8$$

from which we obtain inequality (14).

E Proof of Theorem 6

The proof is similar to the security proof of PSS in [4]. The only difference is that we simulate a decryption oracle as in theorem 3. This adds an error probability of $q_D \cdot 2^{-k_1}$.