This is a revised version of the extended abstract "RSA–OAEP is Secure under the RSA Assumption" which appeared in Advances in Cryptology – Proceedings of CRYPTO '2001 (19 – 23 august 2001, Santa Barbara, California, USA) J. Kilian Ed. Springer-Verlag, LNCS 2139, pages 260–274.

RSA–OAEP is Secure under the RSA Assumption*

Eiichiro Fujisaki[1], Tatsuaki Okamoto[1], David Pointcheval[2], and Jacques Stern[2]

[1] NTT Labs, 1-1 Hikarino-oka, Yokosuka-shi, 239-0847 Japan.
E-mail: {fujisaki,okamoto}@isl.ntt.co.jp.
[2] Dépt d'Informatique, ENS – CNRS, 45 rue d'Ulm, 75230 Paris Cedex 05, France.
E-mail: {David.Pointcheval,Jacques.Stern}@ens.fr
URL: http://www.di.ens.fr/users/{pointche,stern}.

Abstract. Recently Victor Shoup noted that there is a gap in the widely-believed security result of OAEP against adaptive chosen-ciphertext attacks. Moreover, he showed that, presumably, OAEP cannot be proven secure from the *one-wayness* of the underlying trapdoor permutation. This paper establishes another result on the security of OAEP. It proves that OAEP offers semantic security against adaptive chosen-ciphertext attacks, in the random oracle model, under the *partial-domain* one-wayness of the underlying permutation. Therefore, this uses a formally stronger assumption. Nevertheless, since partial-domain one-wayness of the RSA function is equivalent to its (full-domain) one-wayness, it follows that the security of RSA–OAEP can actually be proven under the sole RSA assumption, although the reduction is not tight.

1 Introduction

The OAEP conversion method [3] was introduced by Bellare and Rogaway in 1994 and was believed to provide semantic security against adaptive chosen-ciphertext attacks [7, 12], based on the one-wayness of a trapdoor permutation, using the (corrected) definition of plaintext-awareness [1].

Victor Shoup [15] recently showed that it is quite unlikely that such a security proof exists — at least for non-malleability — under the one-wayness of the permutation. He also proposed a slightly modified version of OAEP, called OAEP+, which can be proven secure, under the one-wayness of the permutation.

Does Shoup's result mean that OAEP is insecure or that it is impossible to prove the security of OAEP? This is a totally misleading view: the result only states that it is highly unlikely to find any proof, under just the one-wayness assumption. In other words, Shoup's result does not preclude the possibility of proving the security of OAEP from stronger assumptions.

This paper uses such a stronger assumption. More precisely, in our reduction, a new computational assumption is introduced to prove the existence of a simulator of the decryption oracle. Based on this idea, we prove that OAEP is semantically secure against adaptive chosen-ciphertext attack in the random oracle model [3], under the *partial-domain* one-wayness of the underlying permutation, which is stronger than the original assumption.

Since partial-domain one-wayness of the RSA function [13] is equivalent to the (full-domain) one-wayness, the security of RSA-OAEP can actually be proven under the one-wayness of the RSA function.

* This revised version improves the reduction cost.

The rest of this paper is organized as follows. Section 2 recalls the basic notions of asymmetric encryption and the various security notions. Section 3 reviews the OAEP conversion [3]. Sections 4 and 5 present our new security result together with a formal proof for general OAEP applications. In Section 6, we focus on the RSA application of OAEP, RSA-OAEP.

2 Public-Key Encryption

The aim of public-key encryption is to allow anybody who knows the public key of Alice to send her a message that only she will be able to recover it through her private key.

2.1 Definitions

A public-key encryption scheme is defined by the three following algorithms:

– The *key generation algorithm* $\mathcal{K}$. On input 1^k, where k is the security parameter, the algorithm $\mathcal{K}$ produces a pair $(\mathsf{pk}, \mathsf{sk})$ of matching public and secret keys. Algorithm $\mathcal{K}$ is probabilistic.
– The *encryption algorithm* $\mathcal{E}$. Given a message m and a public key pk, $\mathcal{E}$ produces a ciphertext c of m. This algorithm may be probabilistic.
– The *decryption algorithm* $\mathcal{D}$. Given a ciphertext c and the secret key sk, $\mathcal{D}$ returns the plaintext m. This algorithm is deterministic.

2.2 Security Notions

The first security notion that one would like for an encryption scheme is *one-wayness*: starting with just public data, an attacker cannot recover the complete plaintext of a given ciphertext. More formally, this means that for any adversary $\mathcal{A}$, her success in inverting $\mathcal{E}$ without the secret key should be negligible over the probability space $\mathcal{M} \times \Omega$, where $\mathcal{M}$ is the message space and Ω is the space of the random coins r used for the encryption scheme, and the internal random coins of the adversary:

$$\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A}) = \Pr_{m,r}[(\mathsf{pk}, sk) \leftarrow \mathcal{K}(1^k) : \mathcal{A}(\mathsf{pk}, \mathcal{E}_{\mathsf{pk}}(m; r)) = m].$$

However, many applications require more from an encryption scheme, namely *semantic security* (*a.k.a. polynomial security* or *indistinguishability of encryptions* [7], denoted IND): if the attacker has some information about the plaintext, for example that it is either "yes" or "no" to a crucial query, no adversary should learn more with the view of the ciphertext. This security notion requires computational impossibility to distinguish between two messages, chosen by the adversary, one of which has been encrypted, with a probability significantly better than one half: her advantage $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A})$, where the adversary $\mathcal{A}$ is seen as a 2-stage Turing machine $(\mathcal{A}_1, \mathcal{A}_2)$, should be negligible, where $\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A})$ is formally defined as.

$$2 \times \Pr_{b,r}\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m_0, m_1, s) \leftarrow \mathcal{A}_1(\mathsf{pk}),\\ c = \mathcal{E}_{\mathsf{pk}}(m_b; r) : \mathcal{A}_2(m_0, m_1, s, c) = b\end{array}\right] - 1.$$

Another notion was defined thereafter, the so-called *non-malleability* [6], in which the adversary tries to produce a new ciphertext such that the plaintexts are meaningfully related. This notion is stronger than the above one, but it is equivalent to semantic security in the most interesting scenario [1].

On the other hand, an attacker can use many kinds of attacks: since we are considering asymmetric encryption, the adversary can encrypt any plaintext of her choice with the public key, hence *chosen-plaintext attack*. She may, furthermore, have access to more information, modeled by partial or full access to some oracles: a plaintext-checking oracle which, on input of a pair (m, c), answers whether c encrypts the message m. This attack has been named the *Plaintext-Checking Attack* [11]; a validity-checking oracle which, on input of a ciphertext c, just answers whether it is a valid ciphertext. This weak oracle (involved in the reaction attacks [8]) had been enough to break some famous encryption schemes [4, 9], namely PKCS #1 v1.5; or the decryption oracle itself, which on the input of any ciphertext, except the challenge ciphertext, responds with the corresponding plaintext (*non-adaptive/adaptive chosen-ciphertext attacks* [10, 12]). The latter, the adaptive chosen-ciphertext attack denoted $\mathsf{CCA2}$, is clearly the strongest one.

A general study of these security notions and attacks was given in [1], we therefore refer the reader to this paper for more details. However, the by now expected security level for public-key encryption schemes is semantic security against adaptive chosen-ciphertext attacks ($\mathsf{IND\text{-}CCA2}$) – where the adversary just wants to distinguish which plaintext, between two messages of her choice, had been encrypted; she can ask any query she wants to a decryption oracle (except the challenge ciphertext). This is the strongest scenario one can define.

3 Review of OAEP

3.1 The Underlying Problems

Consider permutation $f : \{0,1\}^k \longrightarrow \{0,1\}^k$, which can also be seen as

$$f : \{0,1\}^{n+k_1} \times \{0,1\}^{k_0} \longrightarrow \{0,1\}^{n+k_1} \times \{0,1\}^{k_0},$$

with $k = n + k_0 + k_1$. In the original description of OAEP from [3], it is only required that f is a trapdoor one-way permutation. However, in the following, we consider two additional related problems: the partial-domain one-wayness and the set partial-domain one-wayness of permutation f:

- (τ, ε)-One-Wayness of f, means that for any adversary $\mathcal{A}$ whose running time is bounded by τ, the success probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$ is upper-bounded by ε, where
$$\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A}) = \Pr_{s,t}[\mathcal{A}(f(s,t)) = (s,t)];$$
- (τ, ε)-Partial-Domain One-Wayness of f, means that for any adversary $\mathcal{A}$ whose running time is bounded by τ, the success probability $\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A})$ is upper-bounded by ε, where
$$\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A}) = \Pr_{s,t}[\mathcal{A}(f(s,t)) = s];$$

– $(\ell, \tau, \varepsilon)$-Set Partial-Domain One-Wayness of f, means that for any adversary $\mathcal{A}$ that outputs a set of ℓ elements within time bound τ, the success probability $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A})$ is upper-bounded by ε, where

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

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 success probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$ (resp. $\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A})$ and $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A})$). The maximum ranges over all adversaries whose running time is bounded by τ. In the third case, there is an obvious additional restriction on this range from the fact that $\mathcal{A}$ outputs sets with ℓ elements. It is clear that for any τ and $\ell \geq 1$,

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

Note that, by randomly selecting an element in the set returned by an adversary to the Set Partial-Domain One-Wayness, one breaks Partial-Domain One-Wayness with probability $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A})/\ell$. This provides the following inequality $\mathsf{Succ}^{\mathsf{pd-ow}}(\tau) \geq \mathsf{Succ}^{\mathsf{s-pd-ow}}(\ell, \tau)/\ell$. However, for specific choices of f, more efficient reductions may exist. Also, in some cases, all three problems are polynomially equivalent. This is the case for the RSA permutation [13], hence the results in section 6.

3.2 The OAEP Cryptosystem

We briefly describe the OAEP cryptosystem $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ obtained from a permutation f, whose inverse is denoted by g. We need two hash functions G and H:

$$G : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{k-k_0} \text{ and } H : \{0,1\}^{k-k_0} \longrightarrow \{0,1\}^{k_0}.$$

Then,

– $\mathcal{K}(1^k)$: specifies an instance of the function f, and of its inverse g. The public key pk is therefore f and the secret key sk is g.
– $\mathcal{E}_{\mathsf{pk}}(m; r)$: given a message $m \in \{0,1\}^n$, and a random value $r \overset{R}{\leftarrow} \{0,1\}^{k_0}$, the encryption algorithm $\mathcal{E}_{\mathsf{pk}}$ computes

$$s = (m \| 0^{k_1}) \oplus G(r) \text{ and } t = r \oplus H(s),$$

and outputs the ciphertext $c = f(s,t)$.
– $\mathcal{D}_{\mathsf{sk}}(c)$: thanks to the secret key, the decryption algorithm $\mathcal{D}_{\mathsf{sk}}$ extracts

$$(s,t) = g(c), \text{ and next } r = t \oplus H(s) \text{ and } M = s \oplus G(r).$$

If $[M]_{k_1} = 0^{k_1}$, the algorithm returns $[M]^n$, otherwise it returns "Reject".

In the above description, $[M]_{k_1}$ denotes the k_1 least significant bits of M, while $[M]^n$ denotes the n most significant bits of M.

4 Security Result

In their paper [3], Bellare and Rogaway provided a security analysis, which proved that the OAEP construction together with any trapdoor one-way permutation is semantically security and (weakly) plaintext-aware. Unfortunately, this just proves semantic security against non-adaptive chosen-ciphertext attacks (*a.k.a.* lunchtime attacks [10] or IND-CCA1). Even if the achieved security was believed to be stronger (namely IND-CCA2), it had never been proven. Thus, Shoup [15] recently showed that it is quite unlikely that such a security proof exists, for any trapdoor one-way permutation. However, he provided a specific proof for RSA with public exponent 3.

In the following, we provide a general security analysis, but under a stronger assumption about the underlying permutation. Indeed, we prove that the scheme is IND-CCA2 in the random oracle model [2], relative to the *partial-domain* one-wayness of function f. More precisely, the following exact security result holds.

Theorem 1. *Let $\mathcal{A}$ be a CCA2–adversary against the "semantic security" of the OAEP conversion $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, with advantage ε and running time t, making q_D, q_G and q_H queries to the decryption oracle, and the hash functions G and H respectively. Then,* $\mathsf{Succ}^{\mathsf{pd-ow}}(t')$ *is greater than*

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

where $t' \leq t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1))$, and T_f denotes the time complexity of function f.

In order to prove this theorem relative to the partial-domain one-wayness of the permutation, one can use the related notion of set partial-domain one-wayness. The theorem follows from the inequalities of the previous section together with the lemma stated below.

Lemma 2. *Let $\mathcal{A}$ be a CCA2–adversary against the "semantic security" of the OAEP conversion $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, with advantage ε and running time t, making q_D, q_G and q_H queries to the decryption oracle, and the hash functions G and H respectively. Then,* $\mathsf{Succ}^{\mathsf{s-pd-ow}}(q_H, t')$ *is greater than*

$$\varepsilon - \frac{2q_D q_G + q_D + q_G}{2^{k_0}} - \frac{2q_D}{2^{k_1}},$$

where $t' \leq t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1))$, and T_f denotes the time complexity of function f.

The next section is devoted to proving this lemma. Hereafter, we will repeatedly use the following simple result:

Lemma 3. *For any probability events* E, F *and* G

$$\Pr[\mathsf{E} \wedge \mathsf{F} \mid \mathsf{G}] \leq \begin{cases} \Pr[\mathsf{E} \mid \mathsf{F} \wedge \mathsf{G}] \\ \Pr[\mathsf{F} \mid \mathsf{G}]. \end{cases}$$

5 Proof of Lemma 2

We prove lemma 2 in three stages. The first presents the reduction of IND-CCA2 adversary $\mathcal{A}$ to algorithm $\mathcal{B}$ for breaking the partial-domain one-wayness of f. Note that, in the present proof, we are just interested in security under the partial-domain one-wayness of f, and not under the full-domain one-wayness of f as in the original paper [3]. The second shows that the decryption oracle simulation employed in this reduction works correctly with overwhelming probability under the partial-domain one-wayness of f. This latter part differs from the original proof [3], and corrects the recently spotted flaw [15]. Finally, we analyze the success probability of our reduction in total, through the incorporation of the above-mentioned analysis of the decryption oracle simulation.

5.1 Description of the Reduction

In this first part, we recall how reduction operates. Let $\mathcal{A} = (A_1, A_2)$ be an adversary against the semantic security of $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, under chosen-ciphertext attacks. Within time bound t, $\mathcal{A}$ asks q_D, q_G and q_H queries to the decryption oracle and the random oracles G and H respectively, and distinguishes the right plaintext with an advantage greater than ε. Let us describe the reduction $\mathcal{B}$.

Top Level Description of the Reduction.

1. $\mathcal{B}$ is given a function f (defined by the public key) and $c^\star \leftarrow f(s^\star, t^\star)$, for $(s^\star, t^\star) \overset{R}{\leftarrow} \{0,1\}^{k-k_0} \times \{0,1\}^{k_0}$. The aim of $\mathcal{B}$ is to recover the partial pre-image $s^\star$ of $c^\star$.
2. $\mathcal{B}$ runs A_1 on the public data, and gets a pair of messages $\{m_0, m_1\}$ as well as state information st. It chooses a random bit b, and then gives $c^\star$ to A_1, as the ciphertext of m_b. $\mathcal{B}$ simulates the answers to the queries of A_1 to the decryption oracle and random oracles G and H respectively. See the description of these simulations below.
3. $\mathcal{B}$ runs $A_2(c^\star, st)$ and finally gets answer b'. $\mathcal{B}$ simulates the answers to the queries of A_2 to the decryption oracle and random oracles G and H respectively. See the description of these simulations below. $\mathcal{B}$ then outputs the partial pre-image $s^\star$ of $c^\star$, if one has been found among the queries asked to H (see below), or the list of queries asked to H.

Simulation of Random Oracles G and H. The random oracle simulation has to simulate the random oracle answers, managing query/answer lists G-List and H-List for the oracles G and H respectively, both are initially set to empty lists:

– for a fresh query γ to G, one looks at the H-List, and for any query δ asked to H with answer H_δ, one builds $z = \gamma \oplus H_\delta$, and checks whether $c^\star = f(\delta, z)$. If for some δ, that relation holds, function f has been inverted, and we can still correctly simulate G, by answering $G_\gamma = \delta \oplus (m_b \| 0^{k_1})$. Note that G_γ is then a uniformly distributed value since $\delta = s^\star$, and the latter is uniformly distributed. Otherwise, one outputs a random value G_γ. In both cases, the pair (γ, G_γ) is concatenated to the G-List.

– for a fresh query δ to H, one outputs a random value H_δ, and the pair (δ, H_δ) is concatenated to the H-List. Note that, once again, for any $(\gamma, G_\gamma) \in$ G-List, one may build $z = \gamma \oplus H_\delta$, and check whether $c^\star = f(\delta, z)$. If for some γ that relation holds, we have inverted the function f.

Simulation of the Decryption Oracle. On query $c = f(s,t)$ to the decryption oracle, decryption oracle simulation $\mathcal{DS}$ looks at each query-answer $(\gamma, G_\gamma) \in$ G-List and $(\delta, H_\delta) \in$ H-List. For each pair taken from both lists, it defines

$$\sigma = \delta, \tau = \gamma \oplus H_\delta, \mu = G_\gamma \oplus \delta,$$

and checks whether

$$c = f(\sigma, \tau) \text{ and } [\mu]_{k_1} = 0^{k_1}.$$

As soon as both equalities hold, $\mathcal{DS}$ outputs $[\mu]^n$. If no such pair is found, "Reject" is returned.

Remarks. When we have found the pre-image of $c^\star$, and thus inverted f, we could output the expected result $s^\star$ and stop the reduction. But for this analysis, we assume the reduction goes on and that $\mathcal{B}$ only outputs it, or the list of queries asked to H, once A_2 has answered b' (or after a time limit).

Even if no answer is explicitly specified, except by a random value for new queries, some are implicitly defined. Indeed, $c^\star$ is defined to be a ciphertext of m_b with random tape $r^\star$:

$$r^\star \leftarrow H(s^\star) \oplus t^\star \text{ and } G(r^\star) \leftarrow s^\star \oplus (m_b \| 0^{k_1}).$$

Since $H(s^\star)$ is randomly defined, $r^\star$ can be seen as a random variable. Let us denote by AskG the event that query $r^\star$ has been asked to G, and by AskH the event that query $s^\star$ has been asked to H. Let us furthermore denote by GBad the event that $r^\star$ has been asked to G, but the answer is something other than $s^\star \oplus (m_b \| 0^{k_1})$ (bit b is fixed in the reduction scenario). Note that the event GBad implies AskG. As seen above, GBad is the only event that makes the random oracle simulation imperfect, in the chosen-plaintext attack scenario. In the chosen-ciphertext attack scenario, we described a decryption simulator that may sometimes fail. Such an event of decryption failure will be denoted by DBad. We thus denote Bad = GBad $\vee$ DBad.

5.2 Notations

In order to proceed to the analysis of the success probability of the above-mentioned reduction, one needs to set up notations. First, we still denote with a star ($\star$) all variables related to the challenge ciphertext $c^\star$, obtained from the encryption oracle. Indeed, this ciphertext, of either m_0 or m_1, implicitly defines hash values, but the corresponding pairs may not appear in the G or H lists. All other variables refer to the decryption query c, asked by the adversary to the decryption oracle, and thus to be decrypted by this simulation. We consider several further events about a ciphertext queried to the decryption oracle:

- CBad denotes the union of the bad events, $\mathsf{CBad} = \mathsf{RBad} \vee \mathsf{SBad}$, where
  - SBad denotes the event that $s = s^\star$;
  - RBad denotes the event that $r = r^\star$, and thus $H(s) \oplus t = H(s^\star) \oplus t^\star$;
- AskRS denotes the intersection of both events about the oracle queries, $\mathsf{AskRS} = \mathsf{AskR} \wedge \mathsf{AskS}$, which means that both r and s have been asked to G and H respectively, since
  - AskR denotes the event that r $(= H(s) \oplus t)$ has been asked to G;
  - AskS denotes the event that s has been asked to H;
- Fail denotes the event that the above decryption oracle simulator outputs a wrong decryption answer to query c. (More precisely, we may denote Fail_i for event Fail on the i-th query c_i $(i = 1, \ldots, q_D)$. For our analysis, however, we can evaluate probabilities regarding event Fail_i in a uniform manner for any i. Hence, we just employ notation Fail.) Therefore, in the global reduction, the event DBad will be set to true as soon as one decryption simulation fails.

Note that the Fail event is limited to the situation in which the plaintext-extractor rejects a ciphertext whereas it would be accepted by the actual decryption oracle. Indeed, as soon as it accepts, we see that the ciphertext is actually valid and corresponds to the output plaintext.

5.3 Analysis of the Decryption Oracle Simulation

We analyze the success probability of decryption oracle simulator $\mathcal{DS}$.

Security Claim. We claim the following, which repairs the previous proof [3], based on the new computational assumption. More precisely, we show that additional cases to consider, due to the corrected definition of plaintext-awareness [1], are very unlikely under the partial-domain one-wayness of the permutation f:

Lemma 4. *When at most one ciphertext $c^\star = f(s^\star, t^\star)$ has been directly obtained from the encryption oracle, but $s^\star$ has not been asked to H, the decryption oracle simulation $\mathcal{DS}$ can correctly produce the decryption oracle's output on query (ciphertext) c $(\neq c^\star)$ with probability greater than ε', within time bound t', where*

$$\varepsilon' \geq 1 - \left(\frac{2}{2^{k_1}} + \frac{2q_G + 1}{2^{k_0}} \right) \text{ and } t' \leq q_G \cdot q_H \cdot (T_f + \mathcal{O}(1)) \,.$$

Before we start the analysis, we recall that the decryption oracle simulator is given the ciphertext c to be decrypted, as well as the ciphertext $c^\star$ obtained from the encryption oracle and both the $\mathsf{G\text{-}List}$ and $\mathsf{H\text{-}List}$ resulting from the interactions with the random oracles G and H. Let us first see that the simulation uniquely defines a possible plaintext, and thus can output the first one it finds. Indeed, with the above definition, several pairs could satisfy the equalities. However, since function f is a permutation, and thus one-to-one, the value of $\sigma = s$

is uniquely defined, and thus δ and H_δ. Similarly, $\tau = t$ is uniquely defined, and thus γ and G_γ: at most one μ may be selected. Then either $[\mu]_{k_1} = 0^{k_1}$ or not.

In the above, one should keep in mind that the G-List and H-List correspond to input-output pairs for the functions G and H. Thus, at most one output is related to a given input.

If the ciphertext has been correctly built by the adversary (r has been asked to G and s to H), the simulation will output the correct answer. However, it will output "Reject" in any other situation, whereas the adversary may have built a valid ciphertext without asking both queries to the random oracles G and H.

Success Probability. Since our goal is to prove the security relative to the partial-domain one-wayness of f, we are only interested in the probability of the event Fail, while ¬AskH occurred, which may be split according to other events. Granted ¬CBad ∧ AskRS, the simulation is perfect, and cannot fail. Thus, we have to consider the complementary events:

$$\Pr[\mathsf{Fail} \mid \neg\mathsf{AskH}] = \Pr[\mathsf{Fail} \wedge \mathsf{CBad} \mid \neg\mathsf{AskH}] + \Pr[\mathsf{Fail} \wedge \neg\mathsf{CBad} \wedge \neg\mathsf{AskRS} \mid \neg\mathsf{AskH}].$$

Concerning the latter contribution to the right hand side, we first note that both

$$\begin{aligned} \neg\mathsf{AskRS} &= \neg\mathsf{AskR} \vee \neg\mathsf{AskS} = (\neg\mathsf{AskR}) \vee (\neg\mathsf{AskS} \wedge \mathsf{AskR}) \\ \neg\mathsf{CBad} &= \neg\mathsf{RBad} \wedge \neg\mathsf{SBad}. \end{aligned}$$

Forgetting ¬AskH for a while, using lemma 3, one gets that $\Pr[\mathsf{Fail} \wedge \neg\mathsf{CBad} \wedge \neg\mathsf{AskRS}]$ is less than

$$\begin{aligned} &\Pr[\mathsf{Fail} \wedge \neg\mathsf{RBad} \wedge \neg\mathsf{AskR}] + \Pr[\mathsf{Fail} \wedge \neg\mathsf{SBad} \wedge (\mathsf{AskR} \wedge \neg\mathsf{AskS})] \\ &\leq \Pr[\mathsf{Fail} \mid \neg\mathsf{AskR} \wedge \neg\mathsf{RBad}] + \Pr[\mathsf{AskR} \mid \neg\mathsf{AskS} \wedge \neg\mathsf{SBad}]. \end{aligned}$$

But without having asked r to G, taking into account the further event ¬RBad, $G(r)$ is unpredictable, and thus the probability that $[s \oplus G(r)]_{k_1} = 0^{k_1}$ is less than 2^{-k_1}. On the other hand, the probability of having asked r to G, without any information about $H(s)$ and thus about r ($H(s)$ not asked, and $s \neq s^\star$, which both come from the conditioning ¬AskS ∧ ¬SBad), is less than $q_G \cdot 2^{-k_0}$. Furthermore, this event is independent of AskH, which yields

$$\Pr[\mathsf{Fail} \wedge \neg\mathsf{CBad} \wedge \neg\mathsf{AskRS} \mid \neg\mathsf{AskH}] \leq 2^{-k_1} + q_G \cdot 2^{-k_0}.$$

We now focus on the former term, Fail ∧ CBad, while ¬AskH, which was missing in the original proof [3] based on a weaker notion of plaintext-awareness. It can be split according to the disjoint sub-cases of CBad, which are SBad and ¬SBad ∧ RBad. Then again using lemma 3,

$$\Pr[\mathsf{Fail} \wedge \mathsf{CBad} \mid \neg\mathsf{AskH}] \leq \Pr[\mathsf{Fail} \mid \mathsf{SBad} \wedge \neg\mathsf{AskH}] + \Pr[\mathsf{RBad} \mid \neg\mathsf{SBad} \wedge \neg\mathsf{AskH}].$$

The latter event means that RBad occurs provided $s \neq s^\star$ and the adversary has not queried $s^\star$ from H. When $s^\star$ has not been asked to H and $s \neq s^\star$, $H(s^\star)$ is unpredictable and independent of $H(s)$ as well as t and $t^\star$. Then, event RBad, $H(s^\star) = H(s) \oplus t \oplus t^\star$, occurs with probability at most 2^{-k_0}.

The former event can be further split according to AskR, and, using once again lemma 3, it is upper-bounded by

$$\Pr[\mathsf{AskR} \mid \mathsf{SBad} \wedge \neg\mathsf{AskH}] + \Pr[\mathsf{Fail} \mid \neg\mathsf{AskR} \wedge \mathsf{SBad} \wedge \neg\mathsf{AskH}].$$

The former event means that r is asked to G whereas $s = s^\star$ and $H(s^\star)$ is unpredictable, thus $H(s)$ is unpredictable. Since r is unpredictable, the probability of this event is at most $q_G \cdot 2^{-k_0}$ (the probability of asking r to G). On the other hand, the latter event means that the simulator rejects the valid ciphertext c whereas $H(s)$ is unpredictable and r is not asked to G. From the one-to-one property of the Feistel network, it follows from $s = s^\star$ that $r \neq r^\star$, and thus $G(r)$ is unpredictable. Then the redundancy cannot hold with probability greater than 2^{-k_1}. To sum up, $\Pr[\mathsf{Fail} \mid \mathsf{SBad} \wedge \neg\mathsf{AskH}] \leq 2^{-k_1} + q_G \cdot 2^{-k_0}$, thus $\Pr[\mathsf{Fail} \wedge \mathsf{CBad} \mid \neg\mathsf{AskH}] \leq 2^{-k_1} + (q_G + 1) \cdot 2^{-k_0}$.

As a consequence,

$$\Pr[\mathsf{Fail} \mid \neg\mathsf{AskH}] \leq \frac{2}{2^{k_1}} + \frac{2q_G + 1}{2^{k_0}}.$$

The running time of this simulator includes just the computation of $f(\sigma, \tau)$ for all possible pairs and is thus bounded by $q_G \cdot q_H \cdot (T_f + \mathcal{O}(1))$.

5.4 Success Probability of the Reduction

This subsection analyzes the success probability of our reduction with respect to the advantage of the $\mathsf{IND\text{-}CCA2}$ adversary. The goal of the reduction is, given $c^\star = f(s^\star, t^\star)$, to obtain $s^\star$. Therefore, the success probability is obtained by the probability that event AskH occurs during the reduction (*i.e.*, $\Pr[\mathsf{AskH}] \leq \mathsf{Succ}^{\mathsf{s\text{-}pd\text{-}ow}}(q_H, t')$, where t' is the running time of the reduction).

We thus evaluate $\Pr[\mathsf{AskH}]$ by splitting event AskH according to event Bad.

$$\Pr[\mathsf{AskH}] = \Pr[\mathsf{AskH} \wedge \mathsf{Bad}] + \Pr[\mathsf{AskH} \wedge \neg\mathsf{Bad}].$$

First let us evaluate the first term.

$$\begin{aligned}
\Pr[\mathsf{AskH} \wedge \mathsf{Bad}] &= \Pr[\mathsf{Bad}] - \Pr[\neg\mathsf{AskH} \wedge \mathsf{Bad}] \\
&\geq \Pr[\mathsf{Bad}] - \Pr[\neg\mathsf{AskH} \wedge \mathsf{GBad}] - \Pr[\neg\mathsf{AskH} \wedge \mathsf{DBad}] \\
&\geq \Pr[\mathsf{Bad}] - \Pr[\mathsf{GBad} \mid \neg\mathsf{AskH}] - \Pr[\mathsf{DBad} \mid \neg\mathsf{AskH}] \\
&\geq \Pr[\mathsf{Bad}] - \Pr[\mathsf{AskG} \mid \neg\mathsf{AskH}] - \Pr[\mathsf{DBad} \mid \neg\mathsf{AskH}] \\
&\geq \Pr[\mathsf{Bad}] - \frac{q_G}{2^{k_0}} - q_D \left(\frac{2}{2^{k_1}} + \frac{2q_G + 1}{2^{k_0}} \right) \\
&\geq \Pr[\mathsf{Bad}] - \frac{2q_D q_G + q_D + q_G}{2^{k_0}} - \frac{2q_D}{2^{k_1}}.
\end{aligned}$$

Here, $\Pr[\mathsf{DBad} \mid \neg\mathsf{AskH}] \leq q_D \left(2 \cdot 2^{-k_1} + (2q_G + 1) \cdot 2^{-k_0}\right)$ is directly obtained from lemma 4, and $\Pr[\mathsf{GBad} \mid \neg\mathsf{AskH}] \leq \Pr[\mathsf{AskG} \mid \neg\mathsf{AskH}]$ is obtained from the fact that event GBad implies AskG. When $\neg\mathsf{AskH}$ occurs, $H(s^\star)$ is unpredictable, and $r^\star = t^\star \oplus H(s^\star)$ is also unpredictable. Hence $\Pr[\mathsf{AskG} \mid \neg\mathsf{AskH}] \le q_G \cdot 2^{-k_0}$.

We then evaluate the second term.

$$\begin{aligned}\Pr[\mathsf{AskH} \wedge \neg\mathsf{Bad}] &\ge \Pr[\mathcal{A} = b \wedge \mathsf{AskH} \wedge \neg\mathsf{Bad}] \\ &= \Pr[\mathcal{A} = b \wedge \neg\mathsf{Bad}] - \Pr[\mathcal{A} = b \wedge \neg\mathsf{AskH} \wedge \neg\mathsf{Bad}].\end{aligned}$$

Here, when $\neg\mathsf{AskH}$ occurs, $H(s^\star)$ is unpredictable, thus $r^\star = t^\star \oplus H(s^\star)$ is unpredictable, and so is b as well. This fact is independent from event $\neg\mathsf{AskH} \wedge \neg\mathsf{Bad}$. In addition,

$$\Pr[\mathsf{Bad}] + (\Pr[\mathsf{AskH} \wedge \neg\mathsf{Bad}] + \Pr[\neg\mathsf{AskH} \wedge \neg\mathsf{Bad}]) = 1.$$

Let $P_A = \Pr[\mathsf{AskH} \wedge \neg\mathsf{Bad}]$, hence

$$\begin{aligned}\Pr[\mathcal{A} = b \wedge \neg\mathsf{AskH} \wedge \neg\mathsf{Bad}] &= \Pr[\neg\mathsf{AskH} \wedge \neg\mathsf{Bad}] \cdot \Pr[\mathcal{A} = b \mid \neg\mathsf{AskH} \wedge \neg\mathsf{Bad}] \\ &= (1 - P_A - \Pr[\mathsf{Bad}]) \cdot \frac{1}{2}.\end{aligned}$$

Furthermore,

$$\Pr[\mathcal{A} = b \wedge \neg\mathsf{Bad}] \ge \Pr[\mathcal{A} = b] - \Pr[\mathsf{Bad}] = \frac{\varepsilon}{2} + \frac{1}{2} - \Pr[\mathsf{Bad}].$$

Therefore,

$$\begin{aligned}P_A = \Pr[\mathsf{AskH} \wedge \neg\mathsf{Bad}] &\ge \frac{\varepsilon}{2} + \frac{1}{2} - \Pr[\mathsf{Bad}] - (1 - P_A - \Pr[\mathsf{Bad}]) \cdot \frac{1}{2} \\ &= \frac{\varepsilon + P_A - \Pr[\mathsf{Bad}]}{2}.\end{aligned}$$

That is, $P_A = \Pr[\mathsf{AskH} \wedge \neg\mathsf{Bad}] \ge \varepsilon - \Pr[\mathsf{Bad}]$.

Combining the evaluation for the first and second terms, one gets

$$\Pr[\mathsf{AskH}] \ge \varepsilon - \frac{2q_Dq_G + q_D + q_G}{2^{k_0}} - \frac{2q_D}{2^{k_1}}.$$

5.5 Complexity Analysis.

Note that during the execution of $\mathcal{B}$, for any new G-query γ, one has to look at all query-answer pairs (δ, H_δ) in the $\mathsf{H\text{-}List}$, and to compute $s = \delta$, $t = \gamma \oplus H_\delta$ as well as $f(s,t)$.

Apparently, one should perform this computation again to simulate the decryption of any ciphertext. Proper bookkeeping allows the computation to be done once for each pair, when the query is asked to the hash functions. Thus, the time complexity of the overall reduction is $t' = t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1))$, where T_f denotes the time complexity for evaluating function f.

6 Application to RSA–OAEP

The main application of OAEP is certainly the famous RSA–OAEP, which has been used to update the PKCS #1 standard [14]. In his paper [15], Shoup was able to repair the security result for a small exponent, $e = 3$, using Coppersmith's algorithm from [5]. However, our result can be applied to repair RSA–OAEP, regardless of the exponent; thanks to the random self-reducibility of RSA, the partial-domain one-wayness of RSA is equivalent to that of the whole RSA problem, as soon as a constant fraction of the most significant bits (or the least significant bits) of the pre-image can be recovered.

We note that, in the original RSA–OAEP [3], the most significant bits are involved in the H function, but in PKCS #1 standards v2.0 and v2.1 [14] and RFC2437, the least significant bits are used: the value $\mathsf{maskedSeed}\|\mathsf{maskedDB}$ is the input to f, the RSA function, where $\mathsf{maskedSeed}$ plays the role of t, and $\mathsf{maskedDB}$ the role of s. But we insist on the fact that the following result holds in both situations (and can be further extended).

One may also remark that the following argument can be applied to any random (multiplicatively) self-reducible problem, such as the Rabin function. Before presenting the final reduction, let us consider the problem of finding small solutions for a linear modular equation.

Lemma 5. *Consider an equation $t + \alpha u = c \bmod N$ which has solutions t and u smaller than 2^{k_0}. For all values of α, except a fraction $2^{2k_0+6}/N$ of them, (t, u) is unique and can be computed within time bound $\mathcal{O}((\log N)^3)$.*

Proof. Consider the lattice

$$L(\alpha) = \{(x, y) \in \mathbb{Z}^2 \mid x - \alpha y = 0 \bmod N\}.$$

We say that $L(\alpha)$ is an ℓ-good lattice (and that α is an ℓ-good value) if there is no non-zero vector of length at most ℓ (with respect to the Euclidean norm). Otherwise, we use the wording ℓ-bad lattices (and ℓ-bad values respectively). It is clear that there are approximately less than $\pi\ell^2$ such ℓ-bad lattices, which we bound by $4\ell^2$. Indeed, each bad value for α corresponds to a point with integer coordinates in the disk of radius ℓ. Thus, the proportion of bad values for α is less than $4\ell^2/N$.

Given an ℓ-good lattice, one applies the Gaussian reduction algorithm. One gets within time $\mathcal{O}((\log N)^3)$ a basis of $L(\alpha)$ consisting of two non-zero vectors U and V such that

$$\|U\| \leq \|V\| \text{ and } |(U, V)| \leq \|U\|^2/2.$$

Let T be the point (t, u), where (t, u) is a solution of the equation $t + \alpha u = c \bmod N$, with both t and u less than 2^{k_0}:

$$T = \lambda U + \mu V, \text{ for some real } \lambda, \mu.$$

$$\begin{aligned}\|T\|^2 = \lambda^2\|U\|^2 + \mu^2\|V\|^2 + 2\lambda\mu(U, V) &\geq (\lambda^2 + \mu^2 - \lambda\mu) \times \|U\|^2 \\ &\geq \left((\lambda - \mu/2)^2 + 3\mu^2/4\right) \times \|U\|^2 \geq 3\mu^2/4 \times \|U\|^2 \geq 3\mu^2\ell^2/4.\end{aligned}$$

Since furthermore we have $\|T\|^2 \leq 2 \times 2^{2k_0}$,

$$|\mu| \leq \frac{2\sqrt{2} \cdot 2^{k_0}}{\sqrt{3} \cdot \ell}, \text{ and } |\lambda| \leq \frac{2\sqrt{2} \cdot 2^{k_0}}{\sqrt{3} \cdot \ell} \text{ by symmetry.}$$

Assuming that we have set from the beginning $\ell = 2^{k_0+2} > 2^{k_0+2}\sqrt{2/3}$, then

$$-\frac{1}{2} < \lambda, \mu < \frac{1}{2}.$$

Choose any integer solution $T_0 = (t_0, u_0)$ of the equation simply by picking a random integer u_0 and setting $t_0 = c - \alpha u_0 \bmod N$. Write it in the basis (U, V): $T_0 = \rho U + \sigma V$ using real numbers ρ and σ. These coordinates can be found, so $T - T_0$ is a solution to the homogeneous equation, and thus indicate a lattice point: $T - T_0 = aU + bV$, with unknown integers a and b. But,

$$T = T_0 + aU + bV = (a + \rho)U + (b + \sigma)V = \lambda U + \mu V,$$

with $-1/2 \leq \lambda, \mu \leq 1/2$. As a conclusion, a and b are the closest integers to $-\rho$ and $-\sigma$ respectively. With a, b, ρ and σ, one can easily recover λ and μ and thus t and u, which are necessarily unique. □

Lemma 6. *Let $\mathcal{A}$ be an algorithm that outputs a q-set containing $k - k_0$ of the most significant bits of the e-th root of its input (partial-domain RSA, for any $2^{k-1} < N < 2^k$, with $k > 2k_0$), within time bound t, with probability ε. There exists an algorithm $\mathcal{B}$ that solves the RSA problem (N, e) with success probability ε', within time bound t' where*

$$\varepsilon' \geq \varepsilon \times (\varepsilon - 2^{2k_0-k+6}),$$
$$t' \leq 2t + q^2 \times \mathcal{O}(k^3).$$

Proof. Thanks to the random self-reducibility of RSA, with part of the bits of the e-th root of $X = (x \cdot 2^{k_0} + r)^e \bmod N$, and the e-th root of $Y = X\alpha^e = (y \cdot 2^{k_0} + s)^e \bmod N$, for a randomly chosen α, one gets both x and y. Thus,

$$(y \cdot 2^{k_0} + s) = \alpha \times (x \cdot 2^{k_0} + r) \bmod N$$
$$\alpha r - s = (y - x\alpha) \times 2^{k_0} \bmod N$$

which is a linear modular equation with two unknowns r and s which is known to have small solutions (smaller than 2^{k_0}). It can be solved using lemma 5.

Algorithm $\mathcal{B}$ just runs twice $\mathcal{A}$, on inputs X and $X\alpha^e$ and next runs the Gaussian reduction on all the q^2 pairs of elements coming from both sets. If the partial pre-images are in the sets, they will be found, unless the random α is bad (*cf.* the Gaussian reduction in lemma 5.) □

Remark 7. The above lemma can be extended to the case where a constant fraction Θ of the leading or trailing bits of the e-th root is found. The reduction runs $1/\Theta$ times the adversary $\mathcal{A}$, and the success probability decreases to approximately $\varepsilon^{1/\Theta}$. Extensions to any constant fraction of consecutive bits are also possible. Anyway, in PKCS #1 v2.0, k_0 is much smaller than $k/2$.

Theorem 8. *Let $\mathcal{A}$ be a CCA2–adversary against the "semantic security" of RSA–OAEP (with a k-bit long modulus, with $k > 2k_0$), with running time bounded by t and advantage ε, making q_D, q_G and q_H queries to the decryption oracle, and the hash functions G and H respectively. Then, the RSA problem can be solved with probability ε' greater than*

$$\varepsilon^2 - 2\varepsilon \cdot \left(\frac{2q_Dq_G + q_D + q_G}{2^{k_0}} + \frac{2q_D}{2^{k_1}} + \frac{32}{2^{k-2k_0}} \right)$$

within time bound $t' \leq 2t + q_H \cdot (q_H + 2q_G) \times \mathcal{O}(k^3)$.

Proof. Lemma 2 states that

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

with $t'' \leq t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1))$, and $T_f = \mathcal{O}(k^3)$. Using the previous results relating q_H-set partial-domain–RSA and RSA, we easily conclude. $\square$

7 Conclusion

Our conclusion is that one can still trust the security of RSA–OAEP, but the reduction is more costly than the original one. However, for other OAEP applications, more care is needed, since the security does not actually rely on the one-wayness of the permutation, only on its partial-domain one-wayness.

Acknowledgments

We thank Victor Shoup, Don Coppersmith and Dan Boneh for fruitful comments.

References

1. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among Notions of Security for Public-Key Encryption Schemes. In *Crypto '98*, LNCS 1462, pages 26–45. Springer-Verlag, Berlin, 1998.
2. M. Bellare and P. Rogaway. Random Oracles Are Practical: a Paradigm for Designing Efficient Protocols. In *Proc. of the 1st CCS*, pages 62–73. ACM Press, New York, 1993.
3. M. Bellare and P. Rogaway. Optimal Asymmetric Encryption – How to Encrypt with RSA. In *Eurocrypt '94*, LNCS 950, pages 92–111. Springer-Verlag, Berlin, 1995.
4. D. Bleichenbacher. A Chosen Ciphertext Attack against Protocols based on the RSA Encryption Standard PKCS #1. In *Crypto '98*, LNCS 1462, pages 1–12. Springer-Verlag, Berlin, 1998.
5. D. Coppersmith. Finding a Small Root of a Univariate Modular Equation. In *Eurocrypt '96*, LNCS 1070, pages 155–165. Springer-Verlag, Berlin, 1996.
6. D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
7. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
8. C. Hall, I. Goldberg, and B. Schneier. Reaction Attacks Against Several Public-Key Cryptosystems. In *Proc. of ICICS'99*, LNCS, pages 2–12. Springer-Verlag, 1999.
9. M. Joye, J. J. Quisquater, and M. Yung. On the Power of Misbehaving Adversaries and Security Analysis of the Original EPOC. In *CT – RSA '2001*, LNCS 2020, pages 208–222. Springer-Verlag, Berlin, 2001.

10. M. Naor and M. Yung. Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *Proc. of the 22nd STOC*, pages 427–437. ACM Press, New York, 1990.
11. T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In *CT – RSA '2001*, LNCS 2020, pages 159–175. Springer-Verlag, Berlin, 2001.
12. C. Rackoff and D. R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *Crypto '91*, LNCS 576, pages 433–444. Springer-Verlag, Berlin, 1992.
13. R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978.
14. RSA Data Security, Inc. Public Key Cryptography Standards – PKCS.
15. V. Shoup. OAEP Reconsidered. In *Crypto '2001*, LNCS 2139, pages 239–259. Springer-Verlag, Berlin, 2001.