RSA–OAEP is Still Alive!

Extended Abstract

(November 27, 2000)

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

[1] NTT Labs, 1-1 Hikarinooka, 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@ens.fr – URL: http://www.di.ens.fr/~pointche.

Abstract. Very recently Victor Shoup showed that the proof of the security (against adaptive chosen ciphertext attack) of OAEP, claimed by Bellare and Rogaway, is incorrect. The flaw in their proof occurs when they simulate the decryption oracle. This paper presents a corrected proof of the security of OAEP, and proves that OAEP is (semantically) secure (against adaptive chosen ciphertext attack) in the random oracle model, under the *partial* one-wayness of the underlying permutation, which is stronger than the originally claimed assumption, one-wayness of the underlying permutation. Since the partial one-wayness of the RSA function is equivalent to its one-wayness, as a result, although the security reduction is not so tight, the security of RSA–OAEP can be proven under the RSA assumption.

1 Introduction

The OAEP conversion was introduced by Bellare and Rogaway in 1994 [3] and it was claimed to provide semantically secure schemes against adaptive chosen ciphertext attacks [4, 6], relative to the one-wayness of a trapdoor permutation, using the (corrected) definition of plaintext-awareness [1].

Very recently, however, Victor Shoup [9] claimed that their proof of the security of OAEP is incorrect. He indicated that their proof technique, based on the "black box" reductions, cannot possibly yield a proof of the security, especially the non-malleability, of OAEP. He also proposed a modified version of OAEP, called OAEP+, which can be proven to be secure.

Does his result mean that OAEP is no more secure, or is it impossible to prove the security of OAEP? No! His result only implies that there is no hope of proving the security of OAEP by the "black box" reduction technique. In other words, his result does not deny the possibility of proving OAEP's security by using another type of reduction technique.

This paper employs a reduction technique different from Bellare-Rogaway's black box reduction: in our reduction, a computational assumption is introduced to prove the existence of a simulator of the decryption oracle. Based on this technique, we prove that OAEP is semantically secure against adaptive chosen ciphertext attack in the random oracle model [3], under the *partial* one-wayness of the underlying permutation, which is stronger than the originally claimed assumption, one-wayness of the underlying permutation. Since the partial one-wayness of the RSA function [7] is equivalent to its one-wayness, the security of RSA-OAEP can be proven under the one-wayness of the RSA function.

2 Recall of OAEP

2.1 The Underlying Problems

Let us consider the 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 OAEP description [3], it is only required to be a trapdoor one-way permutation. However, in the following, we consider two kinds of problems, the one-wayness of f and the partial one-wayness of f:

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

We denote by $\mathsf{Succ}^{\mathsf{ow}}(\tau)$, (resp. $\mathsf{Succ}^{\mathsf{p-ow}}(\tau)$) the maximal success probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$ (resp. $\mathsf{Succ}^{\mathsf{p-ow}}(\mathcal{A})$) over all the adversaries whose running times are bounded by τ.

2.2 The OAEP Cryptosystem

Let us now consider the OAEP conversion $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ obtained from this permutation f, whose inverse is denoted by g. We need G and H, two hash functions:

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

Then,

- $\mathcal{K}(1^k)$: specifies such a function f, and its inverse g. The public key pk is therefore the function f and the secret key sk is the function g.
- $\mathcal{E}_{\mathsf{pk}}(m)$: on any message $m \in \{0,1\}^n$, and a random value $r \stackrel{R}{\leftarrow} \{0,1\}^{k_0}$, one computes
$$s = m \| 0^{k_1} \oplus G(r) \text{ and } t = r \oplus H(s),$$
then the ciphertext is $c = f(s,t)$.
- $\mathcal{D}_{\mathsf{sk}}(c)$: thanks to the secret key, one extracts $(s,t) = g(c)$, then
$$r = t \oplus H(s) \text{ and } M = s \oplus G(r),$$
eventually, if $[M]_{k_1} = 0^{k_1}$, then it returns $[M]^n$, otherwise it returns "Reject".

In 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.

3 Security Result

Let us turn to the security analysis. Indeed, we want to prove that this scheme is IND-CCA in the random oracle model [2], relative to the (partial) one-wayness of the function f. More precisely, one can claim the following exact security result.

Theorem 1. *For any CCA–adversary $\mathcal{A}$ against the "semantic security" of $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, within a time bounded by t, after q_D, q_G and q_H queries to the decryption oracle, and the hash functions G and H respectively, his advantage ε is upper-bounded by*

$$\frac{2}{1 - \frac{q_G}{2^{k_0}} - \frac{q_H}{2^{k-k_0}}} \times \left(\mathsf{Succ}^{\mathsf{ow}}(t') + q_H \times \mathsf{Succ}^{\mathsf{p-ow}}(t') + q_D \times \left(\frac{2}{2^{k_1}} + \frac{q_G + 1}{2^{k_0}} \right) + \frac{q_G}{2^{k-1}} \right),$$

where

$$t' \leq t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1)),$$

and T_f denotes the time complexity of the function f.

4 Proof of the Theorem

For proving this theorem, we will proceed in two stages. The first one considers the classical semantic security of OAEP against chosen-plaintext attacks, under the one-wayness of the permutation f. We then provide a simulator for the decryption oracle, which is correct with an overwhelming probability under the partial one-wayness of permutation f. This latter part differs from the original proof [3], and corrects the flaw recently spotted [9].

4.1 Semantic Security

Thus, let us first claim the classical result about the IND-CPA of OAEP.

Lemma 2. *Let us consider a CPA–adversary $\mathcal{A}$ against the "semantic security" of $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, within a time bounded by t, with advantage ε, after q_G and q_H queries to the hash functions G and H respectively. Then there exists an adversary $\mathcal{B}$ against the one-wayness of the permutation f, with a success probability ε', within a time bound t' where*

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

Proof. The semantic security of this scheme intuitively comes from the fact that for any adversary, in order to have any information about the encrypted message m, she must have asked the query r to G. But in order to have any information about the correct r, she must have asked s to H. Then the r and the s can be found in the queries asked to G and H, then $t = r \oplus H(s)$. The pair (s, t) provides the pre-image of the ciphertext c by the function f (hence the inversion of f.) This part is similar to the original one [3].

Formalization. We thus describe a simulator $\mathcal{B}$ which provides all the view of the adversary as if she were in a real attack. Let us consider $\mathcal{A} = (A_1, A_2)$, an adversary against the semantic security of $(\mathcal{K}, \mathcal{E}, \mathcal{D})$, using a chosen-plaintext attack. Within time bound t, she asks q_G and q_H queries to the hash functions G and H respectively, and distinguishes the right plaintext with an advantage greater than ε. Let us describe the simulator $\mathcal{B}$:

1. $\mathcal{B}$ first runs $\mathcal{K}(1^k)$ to get a function f (defined by the public key);
2. $\mathcal{B}$ is then given $c^\star \leftarrow f(s^\star, t^\star)$, for $(s^\star, t^\star) \stackrel{R}{\leftarrow} \{0,1\}^{k-k_0} \times \{0,1\}^{k_0}$. $\mathcal{B}$ is aimed to recover the pre-image $(s^\star, t^\star)$.
3. $\mathcal{B}$ runs A_1 on the public data, and gets a pair of messages $\{m_0, m_1\}$ as well as a state information st. It chooses a random bit b, and then outputs $c^\star$, as the ciphertext of m_b.
4. $\mathcal{B}$ runs $A_2(c^\star, st)$ and finally gets an answer b'. Then $\mathcal{B}$ outputs the pre-image of $c^\star$, if one has been found among the queries asked to G and H (see below), or Fail.

Of course, during all this simulation, the simulator also has to simulate the random oracle answers, managing query/answer lists G-List and H-List for the oracles G and H respectively, both initially set to empty lists:

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

When we have found the pre-image of $c^\star$, and thus inverted f, one could output it and stop the game. But for the analysis, we assume the game follows and $\mathcal{B}$ only outputs the answer (or Fail, if no pre-image has been found) when A_2 answers b'.

All this simulation is perfect, namely from the random oracle point of view. Indeed, as we have seen, a new uniformly distributed value is returned for any new query. However, the function property of the random oracles may fail: if two distinct answers are given for a same query. Even if no answer is explicitly specified, excepted by a random value for new queries, some are implicitly defined. Indeed, $c^\star$ is defined to be a ciphertext of m_b with a 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 the query $r^\star$ has been asked to G, and by AskH the event that the query $s^\star$ has been asked to H, and more specifically by FAskH

the event that the query $s^\star$ has been asked to H during the find-stage (by A_1). Let us furthermore denote

- by FBad the event that $r^\star$ has been queried to G in the find-stage (by A_1), but answered by something else than $s^\star \oplus m_0 \| 0^{k_1}$ or $s^\star \oplus m_1 \| 0^{k_1}$;
- and by GBad the event that the query $r^\star$ has been asked to G in the guess-stage (by A_2), but answered by something else than $s^\star \oplus m_0 \| 0^{k_1}$ or $s^\star \oplus m_1 \| 0^{k_1}$. Furthermore when this query has been asked, $H(s^\star)$ has not been asked before (otherwise the simulation of G is correct, because of the control during the G-simulation presented above: $G_\gamma \leftarrow \delta \oplus m_b \| 0^{k_1} = s^\star \oplus m_b \| 0^{k_1}$.)

Note that both events FBad and GBad imply AskG. As seen above, FBad and GBad are the only events which make the simulation not to be perfect, then let us denote

$$\mathsf{Bad} = \mathsf{FBad} \vee \mathsf{GBad}.$$

Probabilities Analysis. We denote by $\Pr[\cdot]$ the probabilities in the real attack, and by $\mathsf{pr}[\cdot]$ the probabilities in the simulated game.

As remarked in the first intuition, because of the randomness of G, and of the way the message is masked by $G(r^\star)$, the adversary cannot gain any advantage, in the real game, without having asked $r^\star$ to G. Indeed the simulation in this case is perfect, since $\neg\mathsf{AskG}$ implies $\neg(\mathsf{FBad} \vee \mathsf{GBad})$, and it is clearly independent of b: m_b only appears when r is remarked to be asked to G. Thus the probability of correctly guessing b is exactly of one half:

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) &= 2 \times \Pr[A = b \,|\, \mathsf{AskG} \wedge \mathsf{AskH}] \times \Pr[\mathsf{AskG} \wedge \mathsf{AskH}] \\
&\quad +2 \times \Pr[A = b \,|\, \mathsf{AskG} \wedge \neg\mathsf{AskH}] \times \Pr[\mathsf{AskG} \wedge \neg\mathsf{AskH}] \\
&\quad +2 \times \Pr[A = b \,|\, \neg\mathsf{AskG}] \times \Pr[\neg\mathsf{AskG}] - 1 \\
&\leq 2 \times \Pr[\mathsf{AskG} \wedge \mathsf{AskH}] + 2 \times \Pr[\mathsf{AskG} \wedge \neg\mathsf{AskH}] \\
&\quad +(2 \times \Pr[A = b \,|\, \neg\mathsf{AskG}] - 1) \\
&\leq 2 \times \Pr[\mathsf{AskG} \wedge \mathsf{AskH}] + 2 \times \Pr[\mathsf{AskG} \wedge \neg\mathsf{AskH}] + 0.
\end{aligned}$$

However, in the following, we are interested in the probabilities in the simulated game, which is perfect unless FBad or GBad happens:

$$\varepsilon \leq 2 \times (\mathsf{pr}[(\mathsf{AskG} \wedge \mathsf{AskH}) \,|\, \neg\mathsf{Bad}] + \mathsf{pr}[(\mathsf{AskG} \wedge \neg\mathsf{AskH}) \,|\, \neg\mathsf{Bad}]) \,.$$

But, one may remark that

$$(\mathsf{AskG} \wedge \neg\mathsf{AskH}) \wedge \neg\mathsf{Bad} = (\mathsf{AskG} \wedge \neg\mathsf{AskH}) \wedge \neg(\mathsf{FBad} \wedge \mathsf{GBad})$$

is exactly the event that $r^\star$ has been asked to G, without having asked $s^\star$ to H. But since we furthermore want $\neg(\mathsf{FBad} \wedge \mathsf{GBad})$, the answer $G(r^\star)$ has been either $s^\star \oplus m_0 \| 0^{k_1}$ or $s^\star \oplus m_1 \| 0^{k_1}$. The probability of such an event is less than

$$q_G \cdot 2^{-k_0} \times 2 \cdot 2^{-k+k_0} = 2q_G \cdot 2^{-k} = q_G \cdot 2^{-k+1}.$$

Therefore,

$$\mathsf{pr}[(\mathsf{AskG} \wedge \mathsf{AskH}) \,|\, \neg\mathsf{Bad}] \geq \varepsilon/2 - q_G \cdot 2^{-k+1}/\mathsf{pr}[\neg\mathsf{Bad}],$$

and thus,

$$\begin{aligned}\mathsf{pr}[\mathsf{AskG} \wedge \mathsf{AskH}] &\geq \mathsf{pr}[(\mathsf{AskG} \wedge \mathsf{AskH}) \wedge \neg\mathsf{Bad}] \\ &\geq \mathsf{pr}[(\mathsf{AskG} \wedge \mathsf{AskH}) \mid \neg\mathsf{Bad}] \times \mathsf{pr}[\neg\mathsf{Bad}] \\ &\geq \varepsilon/2 \times \mathsf{pr}[\neg\mathsf{Bad}] - q_G \cdot 2^{-k+1}.\end{aligned}$$

Therefore, we just have to evaluate the probability $\mathsf{pr}[\mathsf{Bad}] = \mathsf{pr}[\mathsf{FBad} \vee \mathsf{GBad}]$:

$$\begin{aligned}\mathsf{pr}[\mathsf{Bad}] &= \mathsf{pr}[\mathsf{Bad} \mid \neg\mathsf{FAskH}] \times \mathsf{pr}[\neg\mathsf{FAskH}] + \mathsf{pr}[\mathsf{Bad} \mid \mathsf{FAskH}] \times \mathsf{pr}[\mathsf{FAskH}] \\ &\leq \mathsf{pr}[\mathsf{FBad} \vee \mathsf{GBad} \mid \neg\mathsf{FAskH}] + \mathsf{pr}[\mathsf{FAskH}].\end{aligned}$$

First, because of the randomness of $s^\star$ (which is uniformly distributed in $\{0,1\}^{k-k_0}$, but not used yet, and thus no information can be revealed about it), the probability of having asked it to H, in the first stage, is less than $q_H/2^{k-k_0}$: $\mathsf{pr}[\mathsf{FAskH}] \leq q_H \cdot 2^{-k+k_0}$.

Furthermore, one has to note that the events FBad or GBad, knowing that $\neg\mathsf{FAskH}$, mean that $\mathcal{A}$ asks $r^\star$ to G without having asked $s^\star$ to H yet. Therefore the random variable $r^\star$ is undefined at the moment of the actual query. Thus FBad or GBad may be set to be true if, later, $H(s^\star)$ is set to a value v such that $v \oplus t^\star$ has been asked to G: $\mathsf{pr}[\mathsf{FBad} \vee \mathsf{GBad} \mid \neg\mathsf{FAskH}] \leq q_G \cdot 2^{-k_0}$. Thus,

$$\mathsf{pr}[\mathsf{Bad}] \leq q_G \cdot 2^{-k_0} + q_H \cdot 2^{-k+k_0}.$$

As a conclusion, the probability that $\mathcal{B}$ outputs the pre-image is greater than

$$\varepsilon/2 \times (1 - q_G \cdot 2^{-k_0} - q_H \cdot 2^{-k+k_0}) - q_G \cdot 2^{-k+1}.$$

Complexity Analysis. Note that during the running time of $\mathcal{B}$, for any new G-query γ, it has to look into all the query-answer $(\delta, H_\delta) \in \mathsf{H\text{-}List}$, and to compute

$$s = \delta, t = \gamma \oplus H_\delta, f(s,t).$$

Furthermore, it needs to do that again to extract the pre-image if AskG and AskH occur. But this can be done only once for each pair, when the query is asked to the hash function. Thus, the time complexity of the overall reduction is

$$t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1)),$$

where T_f denotes the time complexity for evaluating the function f. □

4.2 Simulation of the Decryption.

If one wants to furthermore consider $\mathcal{A}$ as a chosen-ciphertext adversary, $\mathcal{B}$ has to be able to simulate the decryption oracle. Let us describe the behavior of a decryption simulator $\mathcal{DS}$. Then, we analyze the success probability of such a simulator.

Description of the Decryption Simulation. On a query $c = f(s,t)$ to the decryption oracle, $\mathcal{DS}$ looks into all the query-answer $(\gamma, G_\gamma) \in$ G-List and $(\delta, H_\delta) \in$ H-List. For each pair of such query-answer, it defines

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

and checks whether for some pair

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

If, for one of them, both equalities hold, then it outputs $[\mu]^n$. Otherwise, "Reject" is returned.

Security Claim. About this simulation, we can claim the following result, which repairs the previous proof [3], thanks to a computational assumption:

Lemma 3. *The decryption simulation part $\mathcal{DS}$, on a ciphertext c, while at most one ciphertext $c^\star$ has been directly obtained from the encryption oracle, can correctly decrypt c with probability greater than ε', within a time bound t', where*

$$\varepsilon' \geq 1 - \left(q_H \times \mathsf{Succ}^{\mathsf{p-ow}}(t') + \frac{2}{2^{k_1}} + \frac{q_G + 1}{2^{k_0}} \right)$$
$$t' \leq q_G \cdot q_H \cdot (T_f + \mathcal{O}(1)) .$$

Before any analysis, let us recall that the plaintext-extractor is given the ciphertext c to be decrypted, as well as the ciphertext $c^\star$ obtained from the decryption oracle and both lists G-List and H-List based on the interactions with the random oracles G and H. Let us first see that this simulation uniquely defines a possible plaintext, and thus can output the first one it finds. Indeed, with above definition, many pairs could make the equalities to be satisfied. But since the function f is one-to-one, the value of $\sigma = s$ is uniquely defined, and thus δ and H_δ. The same way, $\tau = t$ 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.

For above remark, eventually note that the lists G-List and H-List correspond to input-output of functions G and H. And thus at most one output is related to a given input.

Furthermore, 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.

Notations. In order to proceed to the analysis of the success probability of above plaintext-extractor, one needs to fix the notations. First, as done in the previous part, we denote with a * all the 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 some hash values, but maybe without appearing in the G nor H lists. All the 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 also have to consider some events:

- as above, we denote by AskH the event that $s^\star$ has been asked to H;
- SBad denotes the event that $s = s^\star$;
- RBad denotes the event that $r = r^\star$, which means that

$$H(s) \oplus t = H(s^\star) \oplus t^\star;$$

- Bad denotes the union of these bad events, $\mathsf{Bad} = \mathsf{RBad} \vee \mathsf{SBad}$;
- AskR denotes the event that r has been asked to G;
- AskS denotes the event that s has been asked to H;
- AskRS denotes the intersection of these events, $\mathsf{AskRS} = \mathsf{AskR} \wedge \mathsf{AskS}$, and thus that both r and s have been asked to G and H respectively;
- Fail denotes the event that the above plaintext-extractor outputs a wrong decryption answer.

Remark that the Fail event is limited to the situation that the plaintext-extractor rejects a ciphertext whereas it would be accepted by the decryption oracle. Indeed, as soon as it accepts, it means that the ciphertext is really valid, with the output plaintext.

Before going further in the analysis, we need to consider a new problem, we call Set Partial-One-Wayness: (k, τ, ν)-Set-Partial-One-Wayness of f, means that for any adversary $\mathcal{A}$, which outputs a set of k elements within a time bound τ, its success probability

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

is upper-bounded by ν. Similarly to the success probabilities for other problems, we define $\mathsf{Succ}^{\mathsf{sp-ow}}(k, \tau)$ to be the maximal success probability $\mathsf{Succ}^{\mathsf{sp-ow}}(\mathcal{A})$ over all the adversaries which output sets of k elements within a time bound τ. However, one may note that, by randomly selecting an element in the set,

$$\frac{1}{k} \times \mathsf{Succ}^{\mathsf{sp-ow}}(k, \tau) \leq \mathsf{Succ}^{\mathsf{p-ow}}(\tau).$$

Success Probability. Clearly, we are interested in the probability of the event Fail, which may be splitted according to other events:

$$\begin{aligned}\mathsf{pr}[\mathsf{Fail}] &= \mathsf{pr}[\mathsf{Fail} \wedge \mathsf{Bad}] + \mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{Bad}] \\ &= \mathsf{pr}[\mathsf{Fail} \wedge \mathsf{Bad}] + \mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{Bad} \wedge \neg\mathsf{AskRS}] + \mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{Bad} \wedge \mathsf{AskRS}]\end{aligned}$$

We can already simplify this expression, since the latter case cannot occur. About the previous one, if one note than

$$\mathsf{pr}[\neg\mathsf{AskRS}] = \mathsf{pr}[\neg\mathsf{AskR} \vee \neg\mathsf{AskS}] = \mathsf{pr}[\neg\mathsf{AskR}] + \mathsf{pr}[\neg\mathsf{AskS} \wedge \mathsf{AskR}],$$

and

$$\mathsf{pr}[\neg\mathsf{Bad}] = \mathsf{pr}[\neg\mathsf{RBad} \wedge \neg\mathsf{SBad}] \leq \min\{\mathsf{pr}[\neg\mathsf{RBad}], \mathsf{pr}[\neg\mathsf{SBad}]\}.$$

then,

$$\begin{aligned}
\mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{Bad} \wedge \neg\mathsf{AskRS}] &= \mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{Bad} \wedge \neg\mathsf{AskR}] \\
&\quad + \mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{Bad} \wedge (\mathsf{AskR} \wedge \neg\mathsf{AskS})] \\
&\leq \mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{RBad} \wedge \neg\mathsf{AskR}] \\
&\quad + \mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{SBad} \wedge (\mathsf{AskR} \wedge \neg\mathsf{AskS})] \\
&\leq \mathsf{pr}[\mathsf{Fail} \mid \neg\mathsf{AskR} \wedge \neg\mathsf{RBad}] \times \mathsf{pr}[\neg\mathsf{AskR} \wedge \neg\mathsf{RBad}] \\
&\quad + \mathsf{pr}[\mathsf{Fail} \mid \mathsf{AskR} \wedge \neg\mathsf{AskS} \wedge \neg\mathsf{SBad}] \\
&\qquad \times \mathsf{pr}[\mathsf{AskR} \wedge \neg\mathsf{AskS} \wedge \neg\mathsf{SBad}] \\
&\leq \mathsf{pr}[\mathsf{Fail} \mid \neg\mathsf{AskR} \wedge \neg\mathsf{RBad}] \\
&\qquad + \mathsf{pr}[\mathsf{AskR} \wedge \neg\mathsf{AskS} \wedge \neg\mathsf{SBad}] \\
&\leq \mathsf{pr}[\mathsf{Fail} \mid \neg\mathsf{AskR} \wedge \neg\mathsf{RBad}] \\
&\quad + \mathsf{pr}[\mathsf{AskR} \mid \neg\mathsf{AskS} \wedge \neg\mathsf{SBad}] \times \mathsf{pr}[\neg\mathsf{AskS} \wedge \neg\mathsf{SBad}] \\
&\leq \mathsf{pr}[\mathsf{Fail} \mid \neg\mathsf{AskR} \wedge \neg\mathsf{RBad}] \\
&\qquad + \mathsf{pr}[\mathsf{AskR} \mid \neg\mathsf{AskS} \wedge \neg\mathsf{SBad}].
\end{aligned}$$

But without having asked r to G, and furthermore $\neg\mathsf{RBad}$, $G(r)$ 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 $s \neq s^\star$), is less than $q_G \cdot 2^{-k_0}$. Therefore

$$\mathsf{pr}[\mathsf{Fail} \wedge \neg\mathsf{Bad} \wedge \neg\mathsf{AskRS}] \leq 2^{-k_1} + q_G \cdot 2^{-k_0}.$$

As a consequence,

$$\mathsf{pr}[\mathsf{Fail}] \leq \mathsf{pr}[\mathsf{Fail} \wedge \mathsf{Bad}] + 2^{-k_1} + q_G \cdot 2^{-k_0}.$$

Now, let us focus on the first term, $\mathsf{Fail} \wedge \mathsf{Bad}$, which was missing in the original proof [3]. We can split it according to AskH:

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

The first part is less than $\mathsf{pr}[\mathsf{AskH}]$, which is upper bounded by $\mathsf{Succ}^{\mathsf{sp-ow}}(q_H, t')$. Indeed, at the end of the game, or after a time bound t' fixed in advance, depending on the expected running time of the reduction, $s^\star$ is one element amongst the q_H queries asked to H, and this set can be returned to break the partial one-wayness of the function f. However, this provides a non-uniform reduction. But one could provide a uniform reduction by stopping the game at one randomly chosen H-query, returning this query as a partial pre-image of $c^\star$ by f.

The second one can be splitted once again according to the disjoint sub-cases of $\mathsf{Bad} = \mathsf{SBad} \vee (\neg\mathsf{SBad} \wedge \mathsf{RBad})$:

$$\begin{aligned}
\mathsf{pr}[\mathsf{Fail} \wedge \mathsf{Bad} \wedge \neg\mathsf{AskH}] &= \mathsf{pr}[\mathsf{Fail} \wedge \mathsf{SBad} \wedge \neg\mathsf{AskH}] \\
&\quad + \mathsf{pr}[\mathsf{Fail} \wedge \mathsf{RBad} \wedge \neg\mathsf{SBad} \wedge \neg\mathsf{AskH}] \\
&\leq \mathsf{pr}[\mathsf{Fail} \mid \mathsf{SBad} \wedge \neg\mathsf{AskH}] + \mathsf{pr}[\mathsf{RBad} \mid \neg\mathsf{SBad} \wedge \neg\mathsf{AskH}]
\end{aligned}$$

The latter event means that RBad occurs provided that $s \neq s^\star$ and the adversary has not asked for $s^\star$ to H. When $s^\star$ is not asked to H and $s \neq s^\star$, $H(s^\star)$ is unpredictable and independent from $H(s)$ 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 means that the simulator rejects the valid ciphertext c, and thus the redundancy should hold whereas $H(s^\star)$ is unknown and unpredictable and $s = s^\star$. Then $H(s)$ is unpredictable, hence r is also unpredictable: the redundancy cannot hold with probability greater than 2^{-k_1}.

As a consequence,

$$\mathsf{pr}[\mathsf{Fail} \wedge \mathsf{Bad}] \leq \mathsf{Succ}^{\mathsf{sp-ow}}(q_H, t') \frac{2}{2^{k_1}} + \frac{q_G + 1}{2^{k_0}}.$$

Thus, the decryption is correctly simulated with probability greater than

$$1 - \left(\mathsf{Succ}^{\mathsf{sp-ow}}(q_H, t') + \frac{2}{2^{k_1}} + \frac{q_G + 1}{2^{k_0}} \right).$$

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

4.3 Combining Semantic Security with the Plaintext-Extractor

After q_D queries to the decryption oracle, all the decryptions will be correctly simulated with probability greater than

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

since the event $\neg$AskH is the same in all the decryption simulations, which is upper-bounded by

$$1 - \mathsf{Succ}^{\mathsf{sp-ow}}(q_H, t') - q_D \times \left(\frac{2}{2^{k_1}} + \frac{q_G + 1}{2^{k_0}} \right).$$

As a consequence, our simulator can make q_D queries to this plaintext extractor, and output the whole pre-image of $c^\star$ with probability greater than

$$\frac{\varepsilon}{2} \times \left(1 - \frac{q_G}{2^{k_0}} - \frac{q_H}{2^{k-k_0}} \right) - \frac{q_G}{2^{k-1}} - \mathsf{Succ}^{\mathsf{sp-ow}}(q_H, t') - q_D \times \left(\frac{2}{2^{k_1}} + \frac{q_G + 1}{2^{k_0}} \right).$$

The running time of the overall simulation is still

$$t' \leq t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1)),$$

because all f computations can be done just once for each new hash query. However, the success probability of our simulator is upper-bounded by $\mathsf{Succ}^{\mathsf{ow}}(t')$, and thus

$$\varepsilon \leq 2 \times \frac{\mathsf{Succ}^{\mathsf{ow}}(t') + \mathsf{Succ}^{\mathsf{sp-ow}}(q_H, t') + q_D \times \left(\frac{2}{2^{k_1}} + \frac{q_G + 1}{2^{k_0}} \right) + \frac{q_G}{2^{k-1}}}{1 - \frac{q_G}{2^{k_0}} - \frac{q_H}{2^{k-k_0}}},$$

where

$$t' \leq t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1)) .$$

But we have furthermore remarked that

$$\mathsf{Succ}^{\mathsf{sp-ow}}(q_H, t') \leq q_H \cdot \mathsf{Succ}^{\mathsf{p-ow}}(t'),$$

hence the theorem.

5 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 [8]. Therefore, in his paper [9], Shoup tried to repair the security result for a small exponent, $e = 3$. However, our result can be applied more efficiently for repairing RSA–OAEP. Indeed, thanks to random self-reducibility of RSA, the partial one-wayness of RSA is equivalent to that of the whole RSA problem:

Lemma 4. *The Partial-RSA problem is equivalent to the RSA problem with respect to probabilistic polynomial time reductions.*

Proof. Thanks to the random self-reducibility of RSA, one can run the algorithm which breaks the partial RSA on $y = (x \cdot 2^{k_0} + r)^e \bmod N$, and then on a related value, but uniformly distributed, $z = \alpha^e \cdot y^{-1} = (x' \cdot 2^{k_0} + r')^e \bmod N$. Then one gets x and x', and thus the relation

$$xx' \cdot 2^{2k_0} + r \cdot (x' 2^{k_0}) + r' \cdot (x 2^{k_0}) + r \cdot r' = \alpha \bmod N.$$

This is a knapsack problem of the form

$$a \cdot U + b \cdot V + W = 1 \bmod N,$$

with unknown values U, V and W which are smaller than 2^{2k_0}.

Using classical techniques on lattice reductions [5], if $2k_0 \leq |N|/4$, (since the lattice is of dimension 4), it can easily work to recover U, V and W which are necessarily r, r' and rr'. Otherwise, one case use some well-known refinements or many relations as above. Anyway, in a usual application of RSA–OAEP, with N over 1024 bits, one can use $k_0 = 128$, which makes the classical technique to apply. □

Theorem 5. *Let us consider a CCA–adversary $\mathcal{A}$ against the "semantic security" of RSA–OAEP, within a time bounded by t, with advantage ε, after q_D, q_G and q_H queries to the decryption oracle, and the hash functions G and H respectively. Then the partial-RSA problem can be solved with probability ε' within a time bound t' where*

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

and

$$t' \leq t + q_G \cdot q_H \cdot (T_f + \mathcal{O}(1)) .$$

Proof. In any case, we have $\mathsf{Succ}^{\mathsf{ow}}(t) \leq \mathsf{Succ}^{\mathsf{p-ow}}(t)$, and thus the success probability of an adversary against RSA–OAEP is less than

$$\frac{2}{1 - \frac{q_G}{2^{k_0}} - \frac{q_H}{2^{k-k_0}}} \times \left((1 + q_H) \times \mathsf{Succ}^{\mathsf{p-ow}}(t') + \frac{2q_D}{2^{k_1}} + \frac{q_D(q_G + 1)}{2^{k_0}} + \frac{q_G}{2^{k-1}} \right).$$

□

6 Conclusion

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

Acknowledgements

We thank Jacques Stern for useful advices on the lattice reduction techniques.

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. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
5. A. Joux and J. Stern. Lattice Reduction: A Toolbox for the Cryptanalyst. *Journal of Cryptology*, 11(3):161–186, 1998.
6. 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.
7. 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.
8. RSA Data Security, Inc. Public Key Cryptography Standards – PKCS.
9. V. Shoup. OAEP Reconsidered. Available on the `ePrint` library (2000/060), November 2000.