A preliminary version of this paper appears in *Proceedings of the Topics in Cryptology - CT-RSA 2009, The Cryptographers' Track at the RSA Conference 2009*, LNCS, 2009. This is the full version.

Strengthening Security of RSA-OAEP

Alexandra Boldyreva[*]

Abstract

OAEP is one of the few standardized and widely deployed public-key encryption schemes. It was designed by Bellare and Rogaway as a scheme based on a trapdoor permutation such as RSA. RSA-OAEP is standardized in RSA's PKCS #1 v2.1 and is part of several standards. RSA-OAEP was shown to be IND-CCA secure in the random oracle model under the standard RSA assumption. However, the reduction is not tight, meaning that the guaranteed level of security is not very high for a practical parameter choice. We first observe that the situation is even worse because the analysis was done in the single-query setting, i.e. where an adversary gets a single challenge ciphertext. This does not take into account the fact that in reality an adversary can observe multiple ciphertexts of related messages. The results about the multi-query setting imply that the guaranteed concrete security can degrade by a factor of q, which is the number of challenge ciphertexts an adversary can get. We re-visit a very simple but not well-known modification of the RSA-OAEP encryption which asks that the RSA function is only applied to a part of the OAEP transform. We show that in addition to the previously shown fact that security of this scheme is tightly related to the hardness of the RSA problem, security does not degrade as the number of ciphertexts an adversary can see increases. Moreover, this scheme can be used to encrypt long messages without using hybrid encryption. We believe that this modification to the RSA-OAEP is easy to implement, and the benefits it provides deserves the attention of standard bodies.

1 Introduction

Background and Motivation. OAEP is one of the few standardized and widely deployed public-key encryption schemes. It was designed by Bellare and Rogaway [5] as a scheme based on a trapdoor permutation such as RSA. RSA-OAEP is standardized in RSA's PKCS #1 v2.1 and is part of the ANSI X9.44, IEEE P1363, ISO 18033-2 and SET standards. The scheme is parameterized by k_0, k_1. The encryption algorithm of OAEP[F] takes a public key f, which is an instance of a trapdoor permutation family F, and a message M, picks k_0-bit string r at random, pads M with k_1 zeros to get M' and computes the ciphertext $C = f(s \parallel t)$ for

[*]School of Computer Science, College of Computing, Georgia Institute of Technology, 266 Ferst Drive, Atlanta, GA 30332, USA. E-mail: `sasha@gatech.edu`. .

$s = G(r) \oplus M'$ and $t = H(s) \oplus r$, where G and H are hash functions. OAEP[F] was proven to be IND-CPA secure assuming F is a one-way trapdoor permutation family [5] and IND-CCA secure assuming F is partial one-way [12], both in the random oracle (RO) model, i.e., where G and H are modeled as random oracles [4]. Partial one-wayness is a stronger property than one-wayness and it asks that given the result of applying a random instance of the function family to a random point x it be hard to compute the first part of x. RSA is believed to be one-way, so under this assumption the result of [5] implies that OAEP[RSA] (RSA-OAEP) is IND-CPA in the RO model. In [12] it was shown that one-waynes of RSA also implies partial one-wayness, therefore RSA-OAEP is IND-CCA under the standard RSA assumption (stating that RSA is one-way), in the RO model.

While the concrete security reduction showing OAEP is IND-CCA secure assuming partial one-wayness of the underlying permutation family is tight, the concrete bound showing RSA-OAEP is IND-CCA under the RSA assumption is quite loose, due to the "lossy" reduction from partial one-wayness to one-wayness of RSA. Such a loose concrete security bound implies that it may be easier to break the scheme than to invert RSA, and to maintain reasonable security guarantees one would need to use the scheme with a larger security parameter. It was shown in [16] that keys of length about 4-5 thousand bits are necessary, i.e. at least 4 times larger than the standard 1024-bit keys, and this means decryption will be about $64 = 4^3$ times slower than before (since decryption requires a modulo exponentiation whose complexity is cubic in the length of the security parameter). This is basically impractical.

Moreover, we note that the definitions of security of encryption in [5, 12] only consider an adversary given a single challenge ciphertext. In reality, of course, an adversary can observe multiple ciphertexts of possibly related messages. Such mismatch was studied in [3, 2], who defined security in the "multi-query" setting where the adversary can see multiple challenge ciphertexts on messages of its choice[1]. The result of [3] implies that security (IND-CPA or IND-CCA) in the single-query setting implies security in the multi-query setting, however, concrete security degrades as the number of queries increases, and this loss cannot be avoided in general. However it is possible for some specific constructions, e.g. [3] shows that IND-CPA security of the ElGamal encryption scheme [11] stays tightly related to security of the decisional Diffie-Hellman problem regardless of how many queries an adversary makes. Concrete security in the multi-query setting of RSA-OAEP has not been explicitly addressed before our work.

Interestingly, an extremely simple modification to the the RSA-OAEP scheme permits several concrete security improvements. Unlike most of alternative constructions that have been suggested [17, 9, 15], the modification we study does not change the transform construction. The modified scheme differs from OAEP in that it uses trapdoor permutations of particular structure. Informally, they just leave the last part of the input (t-part of the output of the OAEP transform) in the clear. The scheme can be immediately instantiated with the RSA family if we apply an RSA function only to the s-part of the OAEP transform output, or to a portion of the s-part. This modification has been suggested under the name OAEP++ by Kobara and Imai in [14] in order to improve concrete security of OAEP. They show that RSA-OAEP++ is IND-CCA secure in the RO model under the standard RSA assumption and the reduction is tight. Moreover, no only the bound in the reduction is significantly improved, but also the running time of the adversary in the reduction. However, they only consider the single-query setting. The result of [3] implies that in the practical multi-query setting the concrete security bound is worse by a factor of q, i.e. security may degrade as an adversary

[1]In fact, [3] considers what they call a "multi-user" setting which also allows the adversary to see multiple challenge ciphertexts under multiple public keys. We do not consider multiple public keys in this work.

observes more ciphertexts of possibly related messages. We note that this modification has been also suggested in [8] for an orthogonal reason of showing some positive results about non-malleability of OAEP when one or both ROs are instantiated with existing functions. The paper [8] neither considers the multi-query setting nor provides concrete security bounds.

Our contributions. We show that this simple modification has even more advantages. We prove that concrete IND-CCA security of the modified RSA-OAEP scheme stays tightly related to one-wayness of RSA regardless of how many challenge ciphertexts an adversary sees (is independent of parameter q). The proof is in Section 5 and it uses the self-reducibility property of RSA. There we explain why does not the same idea apply to the original RSA-OAEP scheme. Hence, the modified RSA-OAEP provides significantly better security guarantees than the original version, for very practical parameter sizes, which results in a very efficient scheme.

Additionally, the modified RSA-OAEP scheme can be used to encrypt long messages without using symmetric encryption in the hybrid encryption construct. For that the function G in the transform is made variable-output-length, i.e. its output size is of the length of the message plus the zero padding of length k_1. For a fixed-output-length hash $G'(\cdot)$ one can efficiently construct $G(\cdot)$ as $G'(\langle 0\rangle \,\|\, \cdot) \,\|\, G'(\langle 1\rangle \,\|\, \cdot) \ldots \| \,G'(\langle l\rangle \,\|\, \cdot)$, where $\langle i\rangle$ means the binary representation of the counter $i \in \mathbb{N}$. The function H in the transform needs to be variable-input-length, which is not a problem. The RSA function is applied to the first k (e.g. 1024 bits) of the s-part of the OAEP transform. The proof of security stays virtually the same. This scheme yields more compact ciphertexts for long messages than the one obtained through the use of hybrid encryption because there is no need to encrypt the symmetric key.

We hope the standard bodies will pay attention to the modified RSA-OAEP as the advantages it offers seem to be well worth a very simple modification to the standard scheme.

More related work. After it was realized by [12] that IND-CCA security of RSA-OAEP is not tight there appeared several alternative encryption schemes using different transforms before applying the RSA function. These include OAEP+ [17], SAEP+ [9], REACT [15]. Another alternative, which was proposed in [18] is the simplest construction and is known as Simple RSA or RSA-KEM. IND-CCA security of all of these schemes are tightly related to the hardness of the RSA problem, in the RO model and in the single-query setting. The latter two schemes, unlike the former two, can also be shown to have an improved security reduction in the multi-query setting (though it was not formally proved). We think it is important to show that the standardized RSA-OAEP scheme has similar properties, with the help of a very simple modification that is easy to implement, because it appears very hard to replace the standardized schemes with completely different constructions.

Improving the concrete security bounds is very important. Many papers besides the aforementioned work of [3] focused on this issue. For example, Coron [10] showed a new proof with improved security reduction for the RSA-based Full-Domain Hash signature scheme and his technique has been widely used since then. Abe et al. [1] improved the time bound in the security proofs of some of RSA-based encryption schemes by considering 4-round Feistel network transformation.

2 Preliminaries

Notation and conventions. We denote by $\{0,1\}^*$ the set of all binary strings of finite length. We will refer to members of $\{0,1\}^*$ as strings. If X, Y are strings then $X \,\|\, Y$ denotes the concatenation of X and Y. If S is a set then $X \xleftarrow{\$} S$ denotes that X is selected uniformly at random from S. If $k \in \mathbb{N}$ then 1^k denotes the string consisting of k consecutive "1" bits. If A is a randomized algorithm and $n \in \mathbb{N}$, then the notation $X \xleftarrow{\$} A(X_1, X_2, \ldots, X_n)$ denotes that X is assigned the outcome of the experiment of running A on inputs $X_1, X_2, \ldots, X_n$. When describing algorithms, if X is a variable and Y is a string, then $X \leftarrow Y$ denotes that X is assigned the value of Y.

All algorithms we consider are possibly randomized unless indicated otherwise. By convention, the running-time of an algorithm is measured relative to the bit-length of the input and refers to both the actual running-time and program size, including that of any overlying experiment, according to some fixed RAM model of computation. k denotes the security parameter. All algorithms we consider run in time polynomial in k.

Syntax of public-key encryption. A public-key encryption (PKE) scheme $\mathcal{PE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with associated message space MsgSp, which may depend on the security parameter k, consists of three algorithms. The key-generation algorithm $\mathcal{K}$ on input 1^k returns a public key pk and matching secret key sk. The encryption algorithm $\mathcal{E}$ takes pk and a plaintext M to return a ciphertext. The deterministic decryption algorithm $\mathcal{D}$ takes sk and a ciphertext C to return a plaintext. The consistency condition requires that for all $k \in \mathbb{N}$ and all $M \in \mathrm{MsgSp}(k)$ the probability of $\mathcal{D}_{sk}(C) = M$ is 1, where the probability is over the experiment

$$(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k) \; ; \; C \xleftarrow{\$} \mathcal{E}_{pk}(M) \; .$$

Security of PKE. We recall the notions of security of public-key encryption (PKE). We only consider the definitions addressing chosen-ciphertext attack (as opposed to a weaker version for chosen-plaintext attack). We present two variants of the standard IND-CCA definition. In the first one the adversary is given a single challenge ciphertext, and in the second definition the adversary can see multiple challenge ciphertexts. We then show the relation between the definitions.

Definition 2.1 [Single- and Multi-query CCA Security of PKE] Let $\mathcal{PE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a PKE scheme. Let the *left or right selector* be the map LR defined by $\mathrm{LR}(M_0, M_1, b) = M_b$ for all equal-length strings M_0, M_1, and for any $b \in \{0,1\}$. For an adversary A and $b \in \{0,1\}$ define the experiment:

```
**Experiment** $\mathbf{Exp}^{\text{ind-cca}}_{\mathcal{PE},A}(1^k)$
    $b \xleftarrow{\$} \{0,1\}$
    $(pk, sk) \xleftarrow{\$} \mathcal{K}(1^k)$
    $d \xleftarrow{\$} A^{\mathcal{E}_{pk}(\mathrm{LR}(\cdot,\cdot,b)), \mathcal{D}_{sk}(\cdot)}$
    If $b = d$ then return 1 else return 0
```

It is mandated the LR encryption oracle (also known as the challenge oracle) is queried on pairs of messages in MsgSp(k) and of *equal* length and the decryption oracle is not queried on the outputs of the LR encryption oracle.

For an adversary A who is allowed to make *a single* query to its challenge oracle (we will refer to such an adversary *a single-query adversary*) define the *single-query(sq)-cca-advantage*, $\mathbf{Adv}^{\text{ind-cca-sq}}_{\mathcal{PE},A}(k)$ as

$$2 \cdot \Pr\left[\mathbf{Exp}^{\text{ind-cca}}_{\mathcal{PE},A}(1^k) = 1 \right] - 1 \; .$$

We define the *multi-query(mq)-cca-advantage*, $\mathbf{Adv}^{\text{ind-cca-mq}}_{\mathcal{PE},A}(k)$ the exact same way, but for the adversary A who can query its challenge oracle an arbitrary number of times. We will refer to such A *a multi-query adversary.*

A scheme $\mathcal{PE}$ is said to be IND-CCA secure in the single- (resp. multi-) query setting if the single-query (resp, multi-query) -cca-advantage of any polynomial-time adversary is negligible. ▮

It is shown by using a hybrid argument in [3] that for any $k \in \mathbb{N}$, a scheme $\mathcal{PE}$ and any multi-query adversary A making q queries to its challenge oracle there exists a single-query adversary B so that

$$\mathbf{Adv}^{\text{ind-cca-mq}}_{\mathcal{PE},A}(k) \quad \leq \quad q \cdot \mathbf{Adv}^{\text{ind-cca-sq}}_{\mathcal{PE},B}(k)\,, \tag{1}$$

where the running time of B is that of A plus $O(\log q)$, and B does the same number of decryption oracle queries as A.

It was also shown in [3] that the above bound is tight and cannot be improved in general. But for specific schemes, e.g. ElGamal, the concrete security in the multi-query setting is basically the same as in the single-query setting.

In this paper we are interested in improving the bound in concrete security treatment of the popular RSA-OAEP scheme in the multi-query setting. Accordingly we recall the computational assumptions used in the analyses of the scheme.

Computational Assumptions. A *trapdoor-permutation generator* is an algorithm $\mathcal{F}$ that on input 1^k returns the description of a permutation and its inverse f, f^{-1}. The trapdoor property means that for every instance f there exist a function f^{-1} with the same domain and range so that $f(f^{-1}) \equiv f^{-1}(f) \equiv \text{ID}$, the identity function.

Definition 2.2 [One-wayness] A trapdoor permutation generator $\mathcal{F}$ is called *one-way* if for every $k \in \mathbb{N}$ and every adversary I its advantage $\mathbf{Adv}^{\text{owf}}_{\mathcal{F},I}(k)$ defined as

$$\Pr\left[\, x = x' \;:\; (f, f^{-1}) \stackrel{\$}{\leftarrow} \mathcal{F}(1^k)\,;\; x \stackrel{\$}{\leftarrow} \{0,1\}^k\,;\; x' \stackrel{\$}{\leftarrow} I(1^k, f, f(x)) \,\right]$$

is negligible. ▮

Definition 2.3 [Partial-Domain One-wayness] A trapdoor permutation generator $\mathcal{F}$ is called *partial-domain one-way* for $k \in \mathbb{N}$ and some extra parameter $k' \leq k$, whch can be a linear function of k, if for every $k \in \mathbb{N}$ and every adversary I its advantage $\mathbf{Adv}^{\text{pd}-\text{owf}}_{\mathcal{F},I}(k, k')$ defined as

$$\Pr\left[\, x[1 \ldots k'] = x' \;:\; (f, f^{-1}) \stackrel{\$}{\leftarrow} \mathcal{F}(1^k)\,;\; x \stackrel{\$}{\leftarrow} \{0,1\}^k\,;\; x' \stackrel{\$}{\leftarrow} I(1^k, f, f(x)) \,\right]$$

is negligible, where $x[1 \ldots k']$ denotes the first k' bits of x. ▮

An *RSA trapdoor permutation generator* is an algorithm $\mathcal{F}$ that on input 1^k returns $(N, e), (N, d)$ where N is the product of two random distinct $\lfloor k/2 \rfloor$-bit primes and $ed \equiv 1 \mod \phi(N)$. (Here $\phi(\cdot)$ is Euler's phi function.)

The standard assumption is that the RSA trapdoor permutation generator is one-way, and the reasonable security level requires k to be at least 1024 bits. It was shown in [12] that under this assumption RSA is also partial one-way. But the concrete reduction in [12] is not tight showing that a much larger RSA modulus is required to guarantee reasonable level of the stronger notion of partial one-wayness.

3 RSA-OAEP and its Security

OAEP ENCRYPTION. The OAEP encryption [5] is parameterized by k_0, k_1 and k_2 (that can be linear functions of k, but typically $k_0 = k_1 = 128$ and $k_2 = k$) and makes use of a trapdoor permutation generator $\mathcal{F}$ with domain and range $\{0,1\}^{k_2}$ and two random oracles

$$G\colon \{0,1\}^{k_0} \to \{0,1\}^{k_2-k_0} \quad \text{and} \quad H\colon \{0,1\}^{k_2-k_0} \to \{0,1\}^{k_0} \ .$$

The message space is $\{0,1\}^{k_2-k_0-k_1}$. The scheme $\text{OAEP}[\mathcal{F}] = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is defined as follows:

- The key generation algorithm $\mathcal{K}(1^k)$ picks a pair $(f, f^{-1}) \overset{\$}{\leftarrow} \mathcal{F}(1^{k_2})$ and returns f as pk and f^{-1} as sk.
- The encryption algorithm $\mathcal{E}(pk, M)$ picks $r \overset{\$}{\leftarrow} \{0,1\}^{k_0}$, computes $s \leftarrow G(r) \oplus (M \parallel 0^{k_1})$, $t \leftarrow H(s) \oplus r$ and $C \leftarrow f(s||t)$ and returns C.
- The decryption algorithm $\mathcal{D}(sk, C)$ computes $s \parallel t \leftarrow f^{-1}(C)$, $r \leftarrow t \oplus H(s)$ and $M \leftarrow s \oplus G(r)$. If the last k_1 bits of M are zeros, then it returns the first $k_2 - k_0 - k_1$ bits of M, otherwise it returns $\perp$.

SECURITY OF OAEP. The encryption scheme $\text{OAEP}[\mathcal{F}]$ is IND-CCA secure in the RO model if the underlying trapdoor permutation generator $\mathcal{F}$ is partial-domain one-way [12]. The concrete security results in [12] are done for the single-query IND-CCA security. We "translate" them into the the multi-query IND-CCA security using the result from [3] recalled in Equation 1.

Theorem 3.1 [12, 3] Let $\mathcal{F}$ be a trapdoor permutation generator with domain and range $\{0,1\}^k$. Let $\text{OAEP}[\mathcal{F}]$ be the encryption scheme defined above. Then for any adversary A making q_e challenge oracle and q_d decryption oracle queries, q_H, q_G queries to RO oracles H and G, there exist an adversary B s.t.

$$\mathbf{Adv}^{\text{pd-owf}}_{\mathcal{F},B}(k, k_2 - k_0) \quad \geq \quad \frac{\mathbf{Adv}^{\text{ind-cca-mq}}_{\text{OAEP}[\mathcal{F}],A}(k)}{2q_e q_H} - \frac{1}{q_e q_H}\left(\frac{q_d q_G + q_d + q_G}{2^{k_0}} + \frac{q_d}{2^{k_1}}\right) ,$$

and the running time of B is that of A plus $q_G \cdot q_H \cdot (T_F(k) + O(1)) + O(\log q_e)$, where $T_F(k)$ is the time needed for evaluating a random instance of $\mathcal{F}$. ▌

As we can see the reduction is not particularly tight, but the situation becomes even worse if we use RSA, pretty much the only practical trapdoor permutation. It is believed to be one-way, and it was shown in [12] that under this assumption it is partial one-way as well, but the reduction is not tight. The concrete result is as follows.

Theorem 3.2 [12, 3] Consider the RSA trapdoor permutation generator with domain and range $\{0,1\}^k$. Let OAEP[RSA] be the encryption scheme defined above. Then for any adversary A making q_e challenge oracle and q_d decryption oracle queries, q_H, q_G queries to RO oracles H and G there exist an adversary B s.t.

$$\begin{aligned}\mathbf{Adv}^{\text{owf}}_{\text{RSA},B}(k) \quad \geq \quad & \frac{(\mathbf{Adv}^{\text{ind-cca-mq}}_{\text{OAEP[RSA]},A}(k))^2}{4q_e} \\ & - \frac{1}{q_e}\left(\frac{q_d q_G + q_d + q_G}{2^{k_0}} + \frac{q_d}{2^{k_1}} + \frac{32}{2^{k-2k_0}}\right) ,\end{aligned}$$

and the running time of B is 2 times that of A plus $q_H \cdot (q_H + 2q_G) \cdot O(k^3) + O(\log q_e)$. ▌

Such a loose concrete security bound implies that to maintain reasonable security guarantees, i.e. so that it not much harder to break the scheme than to invert 1024-bit RSA, one would need to use the scheme with a larger security parameter. It is shown in [16] show that keys of length about 4-5 thousand bits are necessary, i.e. at least 4 times larger that the standard 1024-bit keys, and this means decryption will be about $64 = 4^3$ times slower than before (since decryption requires a modulo exponentiation whose complexity is cubic in the length of the parameters). This is basically impractical. Note that the this estimate is for $q_e = 1$, i.e. when a single challenge ciphertext is considered. If we take into account the maximum number of queries to the challenge oracle an adversary makes – q_e, then to have reasonable security guarantees in the practical multi-query settings the RSA parameters should be even larger, making the scheme's algorithms prohibitively slow.

4 Known Concrete Security Improvements

Interestingly an extremely simple modification to the scheme permits several concrete security improvements. The modified scheme differs from OAEP[F] in that it uses trapdoor permutations of particular structure, which leave the last part of the input in the clear. Let $\mathcal{F}$ be a generator producing trapdoor permutations with domain and range $\{0,1\}^k$. Define a new generator $\mathcal{F}_k$ first to run $\mathcal{F}$; let (f, f^{-1}) be its output of $\mathcal{F}$, and define the first output of $\mathcal{F}_k$ as $f_p(x) \equiv f(x[1,\ldots,k]) \parallel \mathrm{ID}(x[k+1,\ldots,p]) = f(x[1,\ldots,k]) \parallel x[k+1,\ldots,p]$ for any inputs x of length $p \geq k$, where $x[1,\ldots,k]$ denotes the first k bits of x. The second output, the inverse permutation, is defined straight-forwardly. With regard to the OAEP construction we will be interested in cases when $p = k_2$ and $k \leq k_2 - k_0$, so that applying $\mathcal{F}_k$ to the output of the OAEP transform leaves the t-part in the clear. This modification has been suggested under the name OAEP++ by Kobara and Imai in [14] in order to improve concrete security of OAEP. This modification has also been previously suggested in [8] for an orthogonal reason of showing some positive results about non-malleability of OAEP when one or both ROs are instantiated with existing functions. The paper [8] does not provide concrete security bounds.

It is basically straightforward to see that if $\mathcal{F}$ is one-way, then $\mathcal{F}_k$ is partial one-way, in that it is infeasible to recover first k bits of the preimage. With respect to RSA, we get that RSA_k, applying RSA to only the first k bits of the input, is partial-one-way under the standard RSA assumption. That immediately implies that OAEP[$\mathcal{F}_k$], when $k \leq k_2 - k_0$ is IND-CCA in the RO model, if $\mathcal{F}$ is one-way, and we get that OAEP[RSA_k] is IND-CCA in the RO model under the standard RSA assumption[2]. For the concrete security result we can use the bound of Theorem 3.1.

But as shown in [14] we can get rid of factor q_h. This is possible for the modified scheme for the following reason. The proof of the original scheme constructs an adversary B breaking partial one-wayess of $\mathcal{F}$ using the IND-CCA adversary A for OAEP[F]. B needs to partially invert its input $y = f(s \parallel t)$, i.e. find s. This input y is given to A as the challenge ciphertext. The proof argues that the only way A can win the IND-CCA game is by querying the random oracle H on s at some point. While B cannot check which of the RO queries A made is the correct value B is looking for (since B does not know the second part t to verify this), it can just pick one query at random. This is where the factor q_h, the number of RO queries, is coming from. For the modified scheme, the proof from [12] applies without a single change,

[2]This was previously observed in [8]. The reduction in [14] does not use this observation and the proof is done "from scratch".

except we can note that B will now be able to select the correct s out of A's RO queries because t is in the clear. B just checks if $f(s_i \parallel t) = y$ for all queries to the random oracle H that A makes. Here is the improved security result, which also takes into account the multi-query setting (not considered in [14]).

In addition, the running time of the constructed adversary in the reduction is also improved. We comment on this below and give more details in the next section.

Theorem 4.1 [14, 3] Let $\mathcal{F}$ be a trapdoor permutation family with domain and range $\{0,1\}^k$. Let $\mathcal{F}_k$ be a trapdoor permutation generator producing permutations with domain and range $\{0,1\}^p$ for $p \geq k$ as defined above. Let $\mathrm{OAEP}[\mathcal{F}_k] = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the encryption scheme defined in Section 3 so that $p = k_2$ and $k \leq k_2 - k_0$. Then for any adversary A making q_e challenge oracle queries, q_d decryption oracle queries, q_H, q_G queries to RO oracles H and G there exist an adversary B s.t.

$$\mathbf{Adv}^{\text{owf}}_{\mathcal{F},B}(k) \quad \geq \quad \frac{\mathbf{Adv}^{\text{ind-cca-mq}}_{\mathrm{OAEP}[\mathcal{F}_k],A}(k)}{2q_e} - \frac{1}{q_e}\left(\frac{q_d q_G + q_d + q_G}{2^{k_0}} + \frac{q_d}{2^{k_1}}\right),$$

and the running time of B is that of A plus $q_d \cdot q_H \cdot (T_{\mathcal{F}_k}(k) + O(1)) + O(\log q_e)$, where $T_{\mathcal{F}_k}(k)$ is the time needed for evaluating a random instance of $\mathcal{F}_k$. ▌

Note an improvement compared to the running time of the adversary in the proof of Theorem 3.1. There the number of trapdoor permutation computations is proportional to $q_G \cdot q_H$ i. Here it is proportional to $q_d \cdot q_H$. This is much better as in practice the number of decrypted ciphertexts can be much smaller that the number of hash computations. We explain the reason for this saving in the next section.

The RSA instantiation result is immediate if we use RSA in place of $\mathcal{F}$ and RSA_k in place of $\mathcal{F}_k$ above.

5 Improving the Security in the Multi-Query Setting

We show that security in the multi-query setting does not have to degrade as more messages are encrypted by each user (when an adversary does multiple queries to the challenge encryption oracle), i.e. we can get rid of the factor q_e in the bound of Theorem 4.1 when $\mathrm{OAEP}(\mathrm{RSA}_k)$ is used. Hence, the modified scheme provides significantly better security guarantees than the original version, for very practical parameter sizes. The following theorem states the improvement result.

Theorem 5.1 Let RSA be a trapdoor permutation generator with domain and range $\{0,1\}^k$. Let RSA_k be a trapdoor permutation generator with domain and range $\{0,1\}^p$ for $p \geq k$ as defined in Section 4. Let $\mathrm{OAEP}[\mathrm{RSA}_k]$ be the encryption scheme defined in Section 3 so that $k_2 = p$ and $k \leq k_2 - k_0$. Then for any adversary A attacking IND-CCA security of the scheme making at most q_e queries to its challenge oracle, q_d decryption oracle queries, q_H, q_G queries to RO oracles H and G, there exist an adversary B s.t.

$$\mathbf{Adv}^{\text{owf}}_{\mathrm{RSA},B}(k) \quad \geq \quad \frac{\mathbf{Adv}^{\text{ind-cca-mq}}_{\mathrm{OAEP}[\mathrm{RSA}_k],A}(k)}{2} - \left(\frac{q_d q_G + q_e q_d + q_e q_G}{2^{k_0}} + \frac{q_d}{2^{k_1}}\right),$$

and the running time of B is that of A plus $(q_e+1)\cdot T_m(k)+(q_d\cdot q_H+1)\cdot T_e(k)+O(\log q_e)$, where $T_m(k)$ and $T_e(k)$ are times required to compute one modulo multiplication and exponentiation respectively in $\mathbb{Z}^*_N$, where $N \leq 2^k$. ▌

What does the improvement mean in practice? The current belief is that 1024-bit RSA provides 80 bits of security, so for any adversary B with reasonable resources $\mathbf{Adv}^{\mathrm{owf}}_{\mathrm{RSA},B}(k) \leq 2^{-80}$ (and there are indications that this estimate is outdated in that it does not take into account newer attacks and growing computing power, and the bound is likely to be lower [13]). Now assume an adversary manages to obtain 2^{20} ciphertexts of chosen messages. This is about the number of TLS connections that were required to mount the well-known attack on RSA-PKCS1 by Bleichenbacher [7] (though his attacks needed that many chosen ciphertexts). Then according to Theorem 4.1 the bound on $\mathbf{Adv}^{\mathrm{ind\text{-}cca\text{-}mq}}_{\mathrm{OAEP[RSA}_k],A}(k)$ is only about 2^{-59}, which not a strong security level. Theorem 5.1 implies that in fact security of the scheme does not degrade as an adversary mounts more chosen-plaintext attacks and stays tightly related to the assumed security level of the underlying RSA problem.

Proof: We show how to modify the proof of security of RSA-OAEP in the single-query setting from [12], which assumes an adversary A attacking IND-CCA security of OAEP[F] (in the RO model). In our case we consider a special case of the scheme, OAEP[RSA$_k$] (i.e. when RSA is applied to the first k bits of the OAEP output, leaving the t-part in the clear). This will allow us to use self-reducibility of RSA and to incorporate the RSA challenge into multiple challenge ciphertexts, which the adversary is allowed to see in the multi-query setting.

Following [12] we use the game-playing technique of [6, 19] and consider a sequence of experiments or games, associated with the adversary A and random oracles $G(\cdot), H(\cdot)$. For the most part the proof is a simple extension of the proof in [12]. The pseudocode for the games is on Figures 1,2 and 3, and the description is below. For $i \in \mathbb{N}$ we let $\Pr[\mathsf{Game}_i]$ denote the probability that Game i outputs 1.

Game 0 corresponds to $\mathbf{Exp}^{\mathrm{ind\text{-}cca}}_{\mathrm{OAEP[RSA}_k],A}(1^k)$, the standard multi-query IND-CCA experiment (c.f. Definition 2.1 for the multi-query adversary). Each of q_e challenge ciphertexts is generated according to the definition of encryption of OAEP[RSA$_k$] as follows. For $1 \leq i \leq q_e$, to encrypt $M_{i,b}$ first r_i^* is chosen at random from $\{0,1\}^{k_0}$. Then $C_i \leftarrow f_k(s_i^* \parallel t_i)$, where $s_i^* = G(r_i^*) \oplus M_{i,b} \parallel 0^{k_1}$ and $t_i = r_i^* \oplus H(s_i^*)$. Decryption oracle queries are answered according to the decryption algorithm of OAEP[RSA$_k$]. By construction and Definition 2.1 we get

$$\frac{1}{2} + \frac{1}{2} \cdot \mathbf{Adv}^{\mathrm{ind\text{-}cca\text{-}mq}}_{\mathrm{OAEP[RSA}_k],A}(k) = \Pr[\mathsf{Game}_0] \; .$$

Game 1 is different from Game 0 in that it moves the computation of the random coins, $r_1^+, \ldots, \ldots, r_{q_e}^+$, used in the challenge ciphertexts explicitly up front, together with the computations of $g_1^+, \ldots, g_{q_e}^+$, the values simulating the corresponding random oracle G answers. By computation we mean choosing the values at random from the corresponding domains ($\{0,1\}^{k_0}$ and $\{0,1\}^{k_2-k_0}$ resp.) and storing the results. Further in the game r_i^+ is used in place of r_i^* and g_i^+ is used in place of $G(r_i^+)$, for all $1 \leq i \leq q_e$. I.e. each challenge ciphertext has the form $f_k(s_i^* \| t_i^*)$, where $s_i^* = (M_{i,b} \parallel 0^{k_1}) \oplus g_i^+$, $t_i^* = h_i^+ \oplus r_i^*$ for $r_i^* = r_i^+$ and $h_i^+ = H(s_i^*)$. And whenever A queries the random oracle G on r_i^+ for any $1 \leq i \leq q_e$, it is given back g_i^+. These changes do not affect the distribution of the view of A compared to that in Game 0, because $(r_1^*, G(r_1^*), \ldots, r_{q_e}^*, G(r_{q_e}^*))$ and $(r_1^+, g_1^+, \ldots, r_{q_e}^+, g_{q_e}^+)$ have the same distribution, since G is a random oracle:

$$\Pr[\mathsf{Game}_1] = \Pr[\mathsf{Game}_0]\ .$$

Game 2 differs from Game 1 only in that the queries to the random oracle G on points $r_1^+, \dots, r_{q_e}^+$ made by the adversary or by the decryption oracle are answered at random independently from the values $g_1^+, \dots, g_{q_e}^+$ used to compute the challenge ciphertexts (e.g. by calling $G(\cdot)$). Hence the challenge ciphertexts are independent from the challenge bit b (since they are uniformly distributed, independent of the rest of A's view) and

$$\Pr[\mathsf{Game}_2] = \frac{1}{2}\ .$$

Similarly to [12] we can argue that the view of A and thus its outputs have the same distribution in Games 1 and 2 unless A queries G oracle on either of the points $r_1^*, \dots, r_{q_e}^*$ (directly or making the decryption oracle make this query). Let us denote the probability of such event in this game $\Pr[\mathsf{AskG}_2]$, and such an event is defined similarly in the following games.

$$\Pr[\mathsf{Game}_2] - \Pr[\mathsf{Game}_1] \le \Pr[\mathsf{AskG}_2]\ .$$

Game 3 is different from Game 2 in that it moves the computation of $s_1^+, \dots, s_{q_e}^+$ and $h_1^+, \dots, h_{q_e}^+$) explicitly up front. By computation we mean choosing the values at random from the corresponding domains ($\{0,1\}^{k_2-k_0}$ and $\{0,1\}^{k_0}$ resp.) and storing the results. Further s_i^+ is used in place of s_i^* and h_i^+ is used in place of $H(s_i^+)$, for all $1 \le i \le q_e$. I.e. each challenge ciphertext has the form $f_k(s_i^*||t_i^*)$, where $s_i^* = s_i^+$, $t_i^* = h_i^+ \oplus r_i^*$ for $r_i^* = r_i^+$ and $h_i^+ = H(s_i^*)$. And whenever A queries the random oracle H on s_i^+ for any $1 \le i \le q_e$, it is given back h_i^+. These changes do not affect the distribution of the view of A compared to that in Game 2, because we replaced each quadruple $(s_i^*, H(s_i^*), g_i^+, b)$ with another having the same distribution, since H is a random oracle:

$$\Pr[\mathsf{AskG}_3] = \Pr[\mathsf{AskG}_2]\ .$$

In Game 4, the difference with Game 3 is only in that the queries to the random oracle H made by the adversary or by the decryption oracle on points $s_1^+, \dots, s_{q_e}^+$ are answered at random independently from the values $h_1^+, \dots, h_{q_e}^+$ used in the challenge ciphertexts (e.g by calling $H(\cdot)$).

Similarly to [12] we can argue that the view of A and thus its outputs have the same distribution in Games 3 and 4 unless A or the decryption oracle queries the H oracle on either of the points $s_1^+, \dots, s_{q_e}^+$. Let's denote the probability of such event in this game $\Pr[\mathsf{AskH}_4]$, and such an event is defined similarly in the following games.

$$\Pr[\mathsf{AskG}_4] - \Pr[\mathsf{AskG}_3] \leq \Pr[\mathsf{AskH}_4]$$

and

$$\Pr[\mathsf{AskG}_4] \leq \frac{q_e(q_G + q_d)}{2^{k_0}} .$$

Game 5 is similar to Game 4 except the way the challenge ciphertexts are generated. In this game they are simply picked at random, independently from everything else. We can argue similarly to the proof in [12] that this does not change the view and the outputs of A. The reason is that f_k is a permutation and in Game 4 it was applied to uniformly distributed points $s_i^* \parallel t_i^*$, where $s_i^* = s_i^+$ and $t_i^* = h_i^+ \oplus r_i^+$.

$$\Pr[\mathsf{AskH}_5] = \Pr[\mathsf{AskH}_4] .$$

Note that the pre-computed values $r_i^+, g_i^+, s_i^+, h_i^+$ for $1 \leq i \leq q_e$ are not used further in the game any more. Therefore in the following games we remove the code generating them for convenience.

Games 6–8 deal with answering decryption oracle queries which were simulated perfectly before that. The definitions of the games and their analysis done in [12] hold for our modified scheme and are independent of the number of the challenge encryption oracle queries A does, but we describe them for completeness. For a k_2-bit ciphertext C we call its last k_0 bits t, and the fist $k_2 - k_0$ bits of $f_k^{-1}(C)$ we call s. We call r the result of xoring t with $H(s)$.

Game 6 is like Game 5 except the decryption oracle rejects all ciphertexts for which the underlying r-value has not been previously queried to the G oracle by the adversary. The views of A in Games 5 and 6 are different only if A queries a valid ciphertext without querying the underlying r-value to G oracle. A ciphertext is valid if the last k_1 bits of $s \oplus G(r)$ are zeros. But if r has not been queried, then $G(r)$ is an independent random string and validity will be satisfied with probability at most 2^{-k_1}. For q_d decryption queries we get

$$\Pr[\mathsf{AskH}_6] - \Pr[\mathsf{AskH}_5] \leq \frac{q_d}{2^{k_1}} .$$

Game 7 is like Game 6 except that the decryption oracle rejects all ciphertexts for which the underlying s-value has not been previously queried to the H oracle by the adversary. The views of A in Games 6 and 7 are different only if A queries a valid ciphertext without querying the underlying s-value to H oracle when the query r was made to the G oracle. Since $r = H(s) \oplus t$, $H(s)$ was not previously defined, it is random and independent. Hence the probability that r was queried is at most $q_G/2^{k_0}$. And for q_d decryption queries we get

$$\Pr[\mathsf{AskH}_7] - \Pr[\mathsf{AskH}_6] \leq \frac{q_d q_G}{2^{k_0}} .$$

In the last Game 8 the decryption oracle queries, for which either of the corresponding r and s values have not been queried, are rejected. The game stores the pairs of the random oracle queries and the corresponding answers in the arrays $\mathsf{G\text{-}list}$ and $\mathsf{H\text{-}list}$. The other ciphertexts are decrypted by using a simple plaintext extractor who expects all previously made G and H queries made by A and returns the matching plaintext. Namely, to decrypt a ciphertext $C \parallel t$, take every $(s_j, H(s_j))$ in $\mathsf{H\text{-}list}$ for $1 \le j \le q_H$, compute $r \leftarrow H(s_j) \oplus t$ and check if there is a pair (r, G_r) for any G_r in the $\mathsf{G\text{-}list}$. If so, check if $f_k(s_j \parallel t) = C \parallel t$ and the last k_1 bits of $s_i \oplus G_r)$ are zeros, then return the rest of $s_i \oplus G_r$. Otherwise, return $\perp$.

The view of the adversary does not change and thus

$$\Pr[\mathsf{AskH}_8] = \Pr[\mathsf{AskH}_7] \ .$$

Putting it all together we get

$$\begin{aligned}
\frac{1}{2} \cdot \mathbf{Adv}^{\text{ind-cca-mq}}_{\text{OAEP}[F_k],A}(k) &= \Pr[\mathsf{Game}_0] - \frac{1}{2} \\
&= \Pr[\mathsf{Game}_1] - \frac{1}{2} \\
&\le \Pr[\mathsf{Game}_2] - \frac{1}{2} + \Pr[\mathsf{AskG}_2] \\
&\le \Pr[\mathsf{AskG}_2] \\
&= \Pr[\mathsf{AskG}_3] \\
&\le \Pr[\mathsf{AskG}_4] + \Pr[\mathsf{AskH}_4] \\
&\le \frac{q_e(q_G + q_d)}{2^{k_0}} + \Pr[\mathsf{AskH}_5] \\
&\le \frac{q_e(q_G + q_d)}{2^{k_0}} + \frac{q_d}{2^{k_1}} + \Pr[\mathsf{AskH}_6] \\
&\le \frac{q_e(q_G + q_d) + q_d q_G}{2^{k_0}} + \frac{q_d}{2^{k_1}} + \Pr[\mathsf{AskH}_7] \\
&= \frac{q_e(q_G + q_d) + q_d q_G}{2^{k_0}} + \frac{q_d}{2^{k_1}} + \Pr[\mathsf{AskH}_8] \ .
\end{aligned}$$

We now claim that there exists an adversary B such that

$$\Pr[\mathsf{AskH}_8] \ \le \ \mathbf{Adv}^{\text{owf}}_{\text{RSA},B}(k) \ . \qquad (2)$$

This is where we use self-reducibility of RSA to improve tightness of the reduction. To justify Equation (2) we construct B as follows. B is given an RSA public key (N, e) and a challenge $y = x^e \mod \mathbb{N}$ for a random $x \in \mathbb{Z}_N^*$. B picks q_e values at random from $\mathbb{Z}_N^*$, let us call them $v_1, \ldots, v_{q_e}$; and q_e values at random from $\{0,1\}^{p-k}$, let us call them $w_1, \ldots, w_{q_e}$. B runs A on public key (N, e), answers its RO queries with random and independent values (and records all queries and answers). To answer the decryption oracle queries B checks if the corresponding G and H queries were made, and in this case a simple plaintext extractor we described above is used; otherwise, the ciphertexts are rejected (B returns $\perp$). For $1 \le i \le q_e$ for an i-th query to the challenge oracle made by A, B returns $(y v_i^e \mod N) \parallel w_i$.

We claim that B simulates the view of A in Game 8 perfectly. (Except for the mismatch between the sets $\mathbb{Z}_{\mathbb{N}}^*$ and $\{0,1\}^k$, which is usually ignored. For the possible simple resolutions of this issue see [12].) The challenge ciphertexts are random and independent strings, and the decryption queries are answered according to the simple plaintext extractor algorithm that uses the recorded queries to the random oracles and the (random) answers. Event AskH_8 means that A made a query h to the random oracle G so that $h[1,\ldots,k]^e = yv_j^e(\mod N)$ for some $1 \le j \le q_e$. B searches for such query and outputs $hv_j^{-1} \mod N$, which is $y^d \mod N$, i.e. it breaks one-wayness of RSA.

We finally justify the running time of B. In addition to running A, to compute each of q_e challenge ciphertexts B does one modulo multiplication (note that under our convention we do not have to count RSA computations in encryption as they are part of the A overlying experiment), to answer each of q_d decryption queries B does one RSA computation for each of q_H pairs stored in $\mathsf{H\text{-}list}$, and does one modulo inverse and multiplication at the end. The reason why the bound in the proof in [12] for the original RSA-OAEP is worse is because there for each decryption oracle query the adversary has to apply RSA to values corresponding to all possible combinations of pairs stored in $\mathsf{G\text{-}list}$ and $\mathsf{H\text{-}list}$ or use more storage to speed up the check. But in any case it stays proportional to $q_G \cdot q_H$. In our case we do not need to test every r-value in $\mathsf{G\text{-}list}$ as the required value can be computed using the t-part of the ciphertext.
▮

Remark. We comment on why does not the above proof showing the security improvement work for the unmodified OAEP[RSA] scheme. The reason is that in the original scheme the RSA permutation is applied to the whole output $s \parallel t$ of the OAEP transform. The tight security of OAEP[RSA] is only shown assuming partial-domain one-wayness of RSA. In the proof above the adversary B given a challenge y could still use self-reducibility of RSA and generate challenge ciphertexts for A as $yv_1^e, \ldots, yv_{q_e}^e \mod N$. In A's view, these ciphertexts have the right distribution (in Game 8) unless A queries the H oracle on any of the underlying s values (the first part of $y^d v_1, \ldots, y^d v_{q_e}$). But if this happens, B cannot compute y^d, as it does not know the remaining part of the transform.

6 Encrypting Long Messages with Modified RSA-OAEP

We observe that the modified RSA-OAEP scheme can be used to encrypt long messages without employing symmetric encryption in the hybrid encryption construct. For that the function G in the transform is made variable-output-length, i.e. it's output is the length of the message plus the zero padding of length k_1. For a fixed-output-length hash $G'(\cdot)$ one can efficiently construct $G(\cdot)$ as $G'(\langle 0 \rangle \parallel \cdot) \parallel G'(\langle 1 \rangle \parallel \cdot) \ldots \parallel G'(\langle l \rangle \parallel \cdot)$, where $\langle i \rangle$ means the binary representation of the counter $i \in \mathbb{N}$. In the RO model G is a random oracle if G' is. The function H in the transform needs to be variable-input-length, which is not a problem, since most of the hash functions are. The RSA function is applied to the first k (e.g. 1024 bits) of the s-part of the OAEP transform. The proof of security stays virtually the same. This scheme yields more compact ciphertexts for long messages than the one obtained through the use of hybrid encryption because there is no need to encrypt the symmetric key.

7 Conclusions

We re-visited a previously suggested slight modification of the well-known and practical RSA-OAEP encryption. We showed that this scheme has extra advantages, namely its IND-CCA security remains tightly related (in the RO model) to hardness of the RSA problem, even in the multi-query setting. Additionally, this scheme can be used for encryption of long messages without employing the hybrid encryption method and symmetric encryption. We believe the modification is very simple to implement and may be considered by the standard bodies.

8 Acknowledgments

We thank David Cash and Eike Kiltz for useful discussions, and Adam O'Neill and anonymous reviewers for their comments on the draft. Alexandra Boldyreva is supported in part by NSF CAREER award 0545659 and NSF Cyber Trust award 0831184.

References

[1] Masayuki Abe, Eike Kiltz, and Tatsuaki Okamoto. Chosen ciphertext security with optimal ciphertext overhead. In *Asiacrypt '08*, volume 5350 of *LNCS*, pages 355–371. Springer-Verlag, Berlin, Germany, 2008.

[2] Olivier Baudron, David Pointcheval, and Jacques Stern. Extended notions of security for multicast public key cryptosystems. In *ICALP*, volume 1853 of *LNCS*, pages 499–511. Springer-Verlag, Berlin, Germany, 2000.

[3] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 259–274, Bruges, Belgium, May 14–18, 2000. Springer-Verlag, Berlin, Germany.

[4] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

[5] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111, Perugia, Italy, May 9–12, 1994. Springer-Verlag, Berlin, Germany.

[6] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426, St. Petersburg, Russia, May 28 – June 1, 2006. Springer-Verlag, Berlin, Germany.

[7] Daniel Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1. In Hugo Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 1–12, Santa Barbara, CA, USA, August 23–27, 1998. Springer-Verlag, Berlin, Germany.

[8] Alexandra Boldyreva and Marc Fischlin. On the security of OAEP. In Kaoru Kurosawa, editor, *ASIACRYPT 2006*, LNCS, pages 210–225. Springer-Verlag, Berlin, Germany, December 2006.

[9] Dan Boneh. Simplified OAEP for the RSA and Rabin functions. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 275–291, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.

[10] Jean-Sébastien Coron. On the exact security of Full Domain Hash. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 229–235, Santa Barbara, CA, USA, August 20–24, 2000. Springer-Verlag, Berlin, Germany.

[11] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[12] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP is secure under the RSA assumption. *Journal of Cryptology*, 17(2):81–104, March 2004.

[13] Burt Kaliski. TWIRL and RSA key size. *RSA Laboratories*, 2003.

[14] Kazukuni Kobara and Hideki Imai. OAEP++ : A very simple way to apply OAEP to deterministic OW-CPA primitives. Cryptology ePrint Archive, Report 2002/130, 2002. `http://eprint.iacr.org/`.

[15] Tatsuaki Okamoto and David Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 159–175, San Francisco, CA, USA, April 8–12, 2001. Springer-Verlag, Berlin, Germany.

[16] David Pointcheval. How to encrypt properly with RSA. *RSA Laboratories' CryptoBytes*, 5(1):9–19, Winter/Spring 2002.

[17] Victor Shoup. OAEP reconsidered. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 239–259, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.

[18] Victor Shoup. A proposal for an ISO standard for public-key encryption. *ISO/IEC JTC 1/SC27*, 2001.

[19] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. cryptology eprint archive, report 2004/332, 2004. `http://eprint.iacr.org/`.

```
Game-0$_A^{G(\cdot),H(\cdot)}(k)$:
  $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
  Run $A$ on input $f_k$ and
    when $G(\cdot)$ is queried on $r$, return $G(r)$
    when $H(\cdot)$ is queried on $s$, return $H(s)$
    when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
      to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
      Pick $r_i^* \xleftarrow{\$} \{0,1\}^{k_0}$
      Compute $s_i^* \leftarrow G(r_i^*) \oplus M_{b,i}||0^{k_1}$
      Compute $C_i^* \leftarrow f_k(s_i^*)$
      Compute $t_i^* \leftarrow H(s_i^*) \oplus r_i^*$
      return $C_i^* \parallel t_i^*$
    when $A$ makes $j$-th query $C_j \parallel t_j$
      to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
      return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
  Until $A$ returns a bit $d$
  Return 1 iff $b = d$
```

```
Game-1$_A^{G(\cdot),H(\cdot)}(k)$:
  $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
  For $1 \le i \le q_e$ pick
    $r_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$, $g_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$
  Run $A$ on input $f_k$ and
    when $G(\cdot)$ is queried on $r$
        and $r = r_l^+$ for some $1 \le l \le q_e$
        then return $g_l^+$ , otherwise return $G(r)$
    when $H(\cdot)$ is queried on $s$, return $H(s)$
    when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
      to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
      Compute $s_i^* \leftarrow g_i^+ \oplus M_{b,i}||0^{k_1}$
      Compute $C_i^* \leftarrow f_k(s_i^*)$
      Compute $t_i^* \leftarrow H(s_i^*) \oplus r_i^+$
      return $C_i^* \parallel t_i^*$
    when $A$ makes $j$-th query $C_j \parallel t_j$
      to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
      return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
  Until $A$ returns a bit $d$
  Return 1 iff $b = d$
```

```
Game-2$_A^{G(\cdot),H(\cdot)}(k)$:
  $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
  For $1 \le i \le q_e$ pick
    $r_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$, $g_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$
  Run $A$ on input $f_k$ and
    when $G(\cdot)$ is queried on $r$
      and $r = r_l^+$ for some $1 \le l \le q_e$
      then return $G(r)$ , otherwise return $G(r)$
    when $H(\cdot)$ is queried on $s$, return $H(s)$
    when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
      to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
      Compute $s_i^* \leftarrow g_i^+ \oplus M_{b,i}||0^{k_1}$
      Compute $C_i^* \leftarrow f_k(s_i^*)$
      Compute $t_i^* \leftarrow H(s_i^*) \oplus r_i^+$
      return $C_i^* \parallel t_i^*$
    when $A$ makes $j$-th query $C_j \parallel t_j$
      to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
      return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
  Until $A$ returns a bit $d$
  Return 1 iff $b = d$
```

```
Game-3$_A^{G(\cdot),H(\cdot)}(k)$:
  $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
  For $1 \le i \le q_e$ pick
    $r_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$, $g_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$
    $s_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$, $h_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$
  Run $A$ on input $f_k$ and
    when $G(\cdot)$ is queried on $r$
      then return $G(r)$
    when $H(\cdot)$ is queried on $s$
        and $s = s_l^+$ for some $1 \le l \le q_e$
        then return $h_l^+$ , otherwise return $H(s)$
    when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
      to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
      Compute $C_i^* \leftarrow f_k(s_i^+)$
      Compute $t_i^* \leftarrow h_i^+ \oplus r_i^+$
      return $C_i^* \parallel t_i^*$
    when $A$ makes $j$-th query $C_j \parallel t_j$
      to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
      return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
  Until $A$ returns a bit $d$
  Return 1 iff $b = d$
```

Figure 1: Games 0–3 for the Proof of Theorem 5.1: Shaded areas indicate the differences between the games.

```
Game-4$_A^{G(\cdot),H(\cdot)}(k)$:
    $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
    For $1 \le i \le q_e$ pick
        $r_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$, $g_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$
        $s_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$, $h_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$
    Run $A$ on input $f_k$ and
        when $G(\cdot)$ is queried on $r$
            then return $G(r)$
        when $H(\cdot)$ is queried on $s$
            and $s = s_l^+$ for some $1 \le l \le q_e$
            then return $H(s)$ , otherwise return $H(s)$
        when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
            to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
            Compute $C_i^* \leftarrow f_k(s_i^+)$
            Compute $t_i^* \leftarrow h_i^+ \oplus r_i^+$
            return $C_i^* \parallel t_i^*$
        when $A$ makes $j$-th query $C_j \parallel t_j$
            to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
            return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
    Until $A$ returns a bit $d$
    Return 1 iff $b = d$
```

```
Game-5$_A^{G(\cdot),H(\cdot)}(k)$:
    $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
    For $1 \le i \le q_e$ pick
        $r_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$, $g_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$
        $s_i^+ \xleftarrow{\$} \{0,1\}^{k_2-k_0}$, $h_i^+ \xleftarrow{\$} \{0,1\}^{k_0}$
    Run $A$ on input $f_k$ and
        when $G(\cdot)$ is queried on $r$
            then return $G(r)$
        when $H(\cdot)$ is queried on $s$
            return $H(s)$
        when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
            to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
            Compute $s_i^* \xleftarrow{\$} \{0,1\}^{k_2-k_0}$ ; $C_i^* \leftarrow f_k(s_i^*)$
            Compute $t_i^* \xleftarrow{\$} \{0,1\}^{k_0}$
            return $C_i^* \parallel t_i^*$
        when $A$ makes $j$-th query $C_j \parallel t_j$
            to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
            return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
    Until $A$ returns a bit $d$
    Return 1 iff $b = d$
```

```
Game-6$_A^{G(\cdot),H(\cdot)}(k)$:
    $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
    Run $A$ on input $f_k$ and
        when $G(\cdot)$ is queried on $r$
            then return $G(r)$
        when $H(\cdot)$ is queried on $s$
            return $H(s)$
        when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
            to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
            Compute $s_i^* \xleftarrow{\$} \{0,1\}^{k_2-k_0}$ ; $C_i^* \leftarrow f_k(s_i^*)$
            Compute $t_i^* \xleftarrow{\$} \{0,1\}^{k_0}$
            return $C_i^* \parallel t_i^*$
        when $A$ makes $j$-th query $C_j \parallel t_j$
            to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
            If $H(f^{-1}(C_j)) \oplus t_j$ was queried to $G(\cdot)$
                then return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
                otherwise return $\perp$
    Until $A$ returns a bit $d$
    Return 1 iff $b = d$
```

```
Game-7$_A^{G(\cdot),H(\cdot)}(k)$:
    $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
    Run $A$ on input $f_k$ and
        when $G(\cdot)$ is queried on $r$
            then return $G(r)$
        when $H(\cdot)$ is queried on $s$
            return $H(s)$
        when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
            to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \le i \le q_e)$
            Compute $s_i^* \xleftarrow{\$} \{0,1\}^{k_2-k_0}$ ; $C_i^* \leftarrow f_k(s_i^*)$
            Compute $t_i^* \xleftarrow{\$} \{0,1\}^{k_0}$
            return $C_i^* \parallel t_i^*$
        when $A$ makes $j$-th query $C_j \parallel t_j$
            to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \le j \le q_d)$
            If $f^{-1}(C_j)$ was queried to $H(\cdot)$
            and $H(f^{-1}(C_j)) \oplus t_j$ was queried to $G(\cdot)$
                then return $\mathcal{D}_{f^{-1}}^{G(\cdot),H(\cdot)}(C_j \parallel t_j)$
                otherwise return $\perp$
    Until $A$ returns a bit $d$
    Return 1 iff $b = d$
```

Figure 2: Games 4-7 for the Proof of Theorem 5.1: Shaded areas indicate the differences between the games.

```
$\mathsf{Game\text{-}8}_A^{G(\cdot),H(\cdot)}(k)$:
    $b \xleftarrow{\$} \{0,1\}$ ; $(f_k, f_k^{-1}) \xleftarrow{\$} \mathcal{K}(1^k)$
    Run $A$ on input $f_k$ and
        when $G(\cdot)$ is queried on $r$
            then return $G(r)$ ; store $(r, G(r))$ in array $\mathsf{G\text{-}list}$
        when $H(\cdot)$ is queried on $s$
            return $H(s)$ ; store $(s, H(s))$ in array $\mathsf{H\text{-}list}$
        when $A$ makes $i$-th query $(M_{0,i}, M_{1,i})$
            to $\mathcal{E}_{f_k}^{G(\cdot),H(\cdot)}(\mathrm{LR}(\cdot,\cdot,b))$, $(1 \leq i \leq q_e)$
            Compute $s_i^* \xleftarrow{\$} \{0,1\}^{k_2-k_0}$ ; $C_i^* \leftarrow f_k(s_i^*)$
            Compute $t_i^* \xleftarrow{\$} \{0,1\}^{k_0}$
            return $C_i^* \parallel t_i^*$
        when $A$ makes $j$-th query $C_j \parallel t_j$
            to $\mathcal{D}_{f_k^{-1}}^{G(\cdot),H(\cdot)}(\cdot)$, $(1 \leq j \leq q_d)$
            If $f^{-1}(C_j)$ was queried to $H(\cdot)$ and $H(f^{-1}(C_j)) \oplus t_j$ was queried to $G(\cdot)$,
            If $\mathsf{G\text{-}list}$ contains $(r, G_r)$ and $\mathsf{H\text{-}list}$ contains $(s, H_s)$, so that
            $H_s \oplus t_j = r,\ f_k(s \parallel t_j) = C_j \parallel t_j$ and the last $k_1$ bits of $s \oplus G_r$ are zeros
            then return the rest of $s \oplus G_r$, otherwise return $\perp$
    Until $A$ returns a bit $d$
    Return 1 iff $b = d$
```

Figure 3: Game 8 for the Proof of Theorem 5.1: Shaded areas indicate the differences between the games.