This is the full version of the extended abstract which appears in
Advances in Cryptology – Proceedings of Asiacrypt '03 (30 november – 4 december 2003, Taiwan)
C. S. Laih Ed. Springer-Verlag, LNCS 2894, pages 1–18.

Chosen-Ciphertext Security without Redundancy

Duong Hieu Phan and David Pointcheval

École normale supérieure – Dépt d'informatique
45 rue d'Ulm, 75230 Paris Cedex 05, France.
{duong.hieu.phan,david.pointcheval}@ens.fr

Abstract. We propose asymmetric encryption schemes for which all ciphertexts are valid (which means here "reachable": the encryption function is not only a probabilistic injection, but also a surjection). We thus introduce the Full-Domain Permutation encryption scheme which uses a random permutation. This is the first IND-CCA cryptosystem based on any trapdoor one-way permutation without redundancy, and more interestingly, the bandwidth is optimal: the ciphertext is over k more bits only than the plaintext, where 2^{-k} is the expected security level. Thereafter, we apply it into the random oracle model by instantiating the random permutation with a Feistel network construction, and thus using OAEP. Unfortunately, the usual 2-round OAEP does not seem to be provably secure, but a 3-round can be proved IND-CCA even without the usual redundancy $m\|0^{k_1}$, under the partial-domain one-wayness of any trapdoor permutation. Although the bandwidth is not as good as in the random permutation model, absence of redundancy is quite new and interesting: many implementation risks are ruled out.

Keywords: asymmetric encryption, semantic security, chosen-ciphertext attack, random oracle model.

1 Introduction

By now, the widely admitted appropriate security level for asymmetric encryption is the so-called *chosen-ciphertext security* (IND-CCA): that is actually the semantic security [16] against adaptive chosen-ciphertext attacks [20]. For achieving semantic security, even in the basic chosen-plaintext scenario, the encryption algorithm must be probabilistic, which means that a given plaintext (with a fixed public key) should be possibly encrypted in many different ways (at least 2^k different ciphertexts if 2^{-k} is the expected security level). This naturally implies an expansion: the ciphertext is at least over k more bits than the plaintext. OAEP achieves the optimal bound if one considers IND-CPA only, but fails when considering IND-CCA [5, 15].

The general idea for designing cryptosystems which are secure in the sense of chosen-ciphertext security is indeed to make the decryption oracle useless by making the creation of new "valid" ciphertexts (which are not produced by actually encrypting some known plaintexts) impossible. The general approach is thus to add some redundancy either to the plaintext before encrypting [5] or in a tag appended to the ciphertext [4, 18]. The former method can be named "encode-then-encrypt", with a randomized bijective encoding (padding), and a trapdoor injective one-way function as encryption [5, 22, 8]. The latter is more like a key-encapsulation technique combined with a MAC of the plaintext, the ciphertext and/or the ephemeral key [10, 1, 18].

For symmetric encryption schemes, Desai [11] avoids the overhead due to the MAC or redundancy by using variable-length input PRF, variable-length output PRF (unbalanced Feistel paradigm) or variable-length input super-PRF (encode-then-encipher). The proposed schemes are chosen-ciphertext secure, without redundancy and the ciphertext expansion is smaller than for any other provably secure scheme.

In the present paper, inspired by this idea (encode-then-encipher), we consider the case of asymmetric encryption, by using a public random permutation which is clearly a bijective encoding, and this leads to the first IND-CCA scheme without any redundancy. More interestingly, the bandwidth of this scheme is optimal.

On the other hand, the security proof holds in the strong and ideal "random permutation model". Such a scheme in a weaker model (the random oracle model or the standard model) would be better.

The second part of this paper is devoted to this goal. We use the construction of OAEP but with 3 rounds, instead of 2, and we can prove that such a scheme is $\mathsf{IND\text{-}CCA}$ and all the ciphertexts are reachable by the encryption algorithm, and are thus valid (or almost all in the most general case).

The rest of the paper is organized as follows: We first briefly recall the security notions for asymmetric encryption; then we present the FDH encryption and we prove that it is $\mathsf{IND\text{-}CCA}$ secure with any trapdoor one-way permutation. Finally we consider the random oracle model, in which we propose a 3-round OAEP for which (almost) any ciphertext is valid (*i.e.*, reachable) and we show that it achieves $\mathsf{IND\text{-}CCA}$ under the partial-domain one-wayness of any trapdoor permutation [15].

2 Public Key Encryption

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

2.1 Definitions

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

– The *key generation algorithm* $\mathcal{G}$. On input 1^k, where k is the security parameter, the algorithm $\mathcal{G}$ produces a pair $(\mathsf{pk}, \mathsf{sk})$ of matching public and private keys.
– The *encryption algorithm* $\mathcal{E}$. Given a message m and a public key pk, $\mathcal{E}_{\mathsf{pk}}(m)$ produces a ciphertext c of m. This algorithm may be probabilistic (involving random coins $r \in \mathcal{R}$, and then denoted $\mathcal{E}_{\mathsf{pk}}(m; r)$.)
– The *decryption algorithm* $\mathcal{D}$. Given a ciphertext c and the secret key sk, $\mathcal{D}_{\mathsf{sk}}(c)$ gives back the plaintext m.

2.2 Security Notions

The widely admitted security notion for encryption schemes is the so-called *semantic security* [16] (a.k.a. *polynomial security/indistinguishability of encryptions*): if the attacker has some *a priori* information about the plaintext, the view of the ciphertext should not increase this information. This security notion requires the computational impossibility to distinguish between two messages, chosen by the adversary itself, which one has been encrypted, with a probability significantly better than one half: its advantage $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A})$, as defined below where the adversary $\mathcal{A}$ is seen as a 2-stage Turing machine (A_1, A_2), should be negligible.

$$\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) = 2 \times \Pr_{b,r}\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{G}(1^k); (m_0, m_1, s) \leftarrow A_1(\mathsf{pk}) \\ c = \mathcal{E}_{\mathsf{pk}}(m_b; r) : A_2(m_0, m_1, s, c) = b\end{array}\right] - 1.$$

Another notion has been thereafter defined, the so-called *non-malleability* [12], but this notion is equivalent to the above one in some specific scenarios [7]. Moreover, it is equivalent to the semantic security [3] in the most interesting scenarios, described below.

Indeed, an attacker can play many kinds of attacks: it may just have access to public data, and then encrypt any plaintext of its choice (*chosen-plaintext attacks*), or have access to extra information, modeled by various oracles. In this model, the strongest oracle is definitely the decryption algorithm, which can be queried on any ciphertext, except the challenge ciphertext (*adaptive/non-adaptive chosen-ciphertext attacks* [17, 20]).

A general study of these security notions and attacks has been driven in [3], we therefore refer the reader to this paper for more details. Actually, one conclusion is that the strongest security level is the so-called *chosen-ciphertext security*, which is the semantic security (IND) under adaptive chosen-ciphertext attacks (CCA), hence the notation $\mathsf{IND\text{-}CCA}$, also known as $\mathsf{IND\text{-}CCA2}$, to be compared to $\mathsf{IND\text{-}CCA1}$, which captures lunchtime attacks [17] only.

2.3 Secure Designs

The expected security level is thus $\mathsf{IND\text{-}CCA}$, which is now required to be provably achieved before any practical use. The last ten years have seen several practical proposals which provide this strong security level. The first, and most famous one, is definitely OAEP [5], a generic conversion proposed by Bellare and Rogaway, which applies to any trapdoor partial-domain one-way permutation, such as RSA, in the random oracle model [15]. Some variants have been recently proposed, which either apply to particular cases (SAEP, SAEP+ [8]) or more general ones (OAEP+ [22]). But they all add some redundancy in the plaintext before encrypting it: a ciphertext that is not properly generated, without knowing the plaintext, is valid with negligible probability only. The latter property had been formally defined by the *plaintext-awareness* notion [5, 3]. Granted it, a decryption oracle does not provide any information.

Some other paddings have also been proposed to apply to more general families of functions, which are not necessarily one-to-one: Fujisaki and Okamoto [13, 14], Pointcheval [19] and Okamoto and Pointcheval [18]. Once again, chosen-ciphertext security is achieved granted redundancy, but in the ciphertext: only properly generated ciphertexts (with some known plaintexts) have a chance to be valid: plaintext-awareness.

3 FDP: Full-Domain Permutation Encryption

In the same vein as the Full-Domain Hash signature [6, 9], we suggest the Full-Domain Permutation encryption, in which one applies a random permutation to the message (and the random coins) before encrypting it with the trapdoor one-way permutation. We therefore obtain the first cryptosystem which achieves chosen-ciphertext security, without redundancy: any ciphertext is valid, and the bandwidth is optimal.

3.1 Description

The FDP-encryption is quite simple, since it uses a random permutation $\mathcal{P}$ (which is a bijective random oracle, or an ideal-cipher with a particular key, say 0. See also [21]). The key generation algorithm selects a trapdoor one-way permutation φ_{pk} (and its inverse ψ_{sk}, granted the trapdoor sk) over $\{0,1\}^{k+\ell}$, and a random permutation $\mathcal{P}$ over the same space —$\{0,1\}^{\ell} \times \{0,1\}^{k}$ is identified to $\{0,1\}^{\ell+k}$. The public key pk thus defines the permutation φ_{pk}, while the private key sk defines the inverse ψ_{sk} of φ_{pk}. Then,

$$\mathcal{E}_{\mathsf{pk}}(m;r) = \varphi_{\mathsf{pk}}(\mathcal{P}(m,r)) \qquad \mathcal{D}_{\mathsf{sk}}(c) = m, \text{ where } (m,r) = \mathcal{P}^{-1}(\psi_{\mathsf{sk}}(c)).$$

The space of the plaintexts is $\{0,1\}^{\ell}$, while the space of the random coins r is $\{0,1\}^{k}$. Note that both $\mathcal{P}$ and $\mathcal{P}^{-1}$ are public permutations.

Note that usual trapdoor one-way permutations are not on a binary set, as it will be discussed in a more extensive way in the following. Anyway, just doubling the computational cost, on average, one easily gets such a particular case from any permutation over an interval: [2] suggested an iterated version.

3.2 Security Result

As already said, the first advantage of this scheme is that any ciphertext is valid: any ciphertext can be decrypted into a plaintext, furthermore any ciphertext can also be reached by the encryption algorithm. The second important advantage comes from the security result given below: it provides chosen-ciphertext security under the intractability of inverting φ, with a security level in 2^{k}, with an overhead of k bits (the random coins). This means that the bandwidth is optimal: contrary to OAEP or OAEP+ which need an overhead of at least $2k$ bits (the random coins and the redundancy), for a similar security level. Of course, this remark only applies to the most general case where $\ell \geq k$ (*e.g.*, $k = 80$ and $k + \ell = 1024$.)

Theorem 1. *Let $\mathcal{A}$ be any chosen-ciphertext adversary against φ-FDP, within time τ. After q_p and q_d queries to the permutation oracles and the decryption oracle respectively,*

$$\mathsf{Adv}_{\pi}^{\mathsf{ind\text{-}cca}}(\mathcal{A}) \leq 2 \times \mathsf{Succ}_{\varphi}^{\mathsf{ow}}(\tau + 2q_p \times T_\varphi) + 2 \times \left(\frac{(q_p + q_d + 1)^2}{2^{k+\ell}} + \frac{q_p}{2^k} + \frac{(q_d + 1)^2}{2^\ell} \right)$$

where T_φ is the time complexity for evaluating φ.

Let us briefly recall that for any algorithm $\mathcal{A}$,

$$\mathsf{Succ}_{\varphi}^{\mathsf{ow}}(\mathcal{A}) = \Pr_{\substack{\mathsf{pk},\\ x \in \{0,1\}^{k+\ell}}} [\mathcal{A}(\varphi_{\mathsf{pk}}(x)) = x], \text{ and } \mathsf{Succ}_{\varphi}^{\mathsf{ow}}(\tau) = \max_{|\mathcal{A}| \leq \tau} \left\{ \mathsf{Succ}_{\varphi}^{\mathsf{ow}}(\mathcal{A}) \right\}.$$

3.3 Sketch of the Proof

The goal of the proof is to simulate the oracles $\mathcal{P}$, $\mathcal{P}^{-1}$, and $\mathcal{D}_{\mathsf{sk}}$ in such a way that the adversary can not distinguish the simulations from the real oracles. In the simulation, the decryption answer for a ciphertext that has not been obtained before is a new random value (and independent with others). We then have to keep the simulation of the random permutation consistent. On the other hand, the challenge is made independent with the plaintexts m_0 and m_1: the adversary has no advantage.

The proof follows by successively modifying the rules involved in the (perfect) simulation presented on the Figure 1, where the oracles $\mathcal{P}$ and $\mathcal{P}^{-1}$ are first simulated by using a perfectly random permutation P and its inverse P^{-1}. The last game provides a simulation of $\mathcal{D}_{\mathsf{sk}}$, without inverting φ_{pk}.

Anyway, the simulation remains almost perfect unless the adversary asks the pre-image via φ_{pk} of the challenge ciphertext to the random permutation $\mathcal{P}^{-1}$: it thus helps to invert φ. The complete proof can be found in the Appendix A.

4 The Random Oracle Model and OAEP

The above result is not so surprising, but the optimal bandwidth is a very good news. However the proof requires a full-domain random permutation, which is hard to find: practical block-ciphers have smaller block sizes. In this section, we present an instantiation of this random permutation, in the random oracle model only. The counter-part will be the need of a stronger assumption about the trapdoor one-way permutation: with a 3-round OAEP, a trapdoor partial-domain one-way permutation leads to an IND-CCA cryptosystem, without redundancy.

4.1 The 2-round OAEP Case

Before studying the 3-round OAEP, let us first consider the more classical 2-round OAEP which can be described as follows: we use two hash functions $\mathcal{G}$ and $\mathcal{H}$ before encrypting with a trapdoor one-way permutation φ_{pk}. More precisely, for encrypting a message m, one randomly chooses r, and computes s and t:

$$s = m \oplus \mathcal{G}(r) \qquad t = r \oplus \mathcal{H}(s).$$

Then, the ciphertext is $c = \varphi_{\mathsf{pk}}(s, t)$. For decryption, one computes

$$(s, t) = \psi_{\mathsf{sk}}(c) \quad r = t \oplus \mathcal{H}(s) \quad m = s \oplus \mathcal{G}(r).$$

The usual way to prove the security of a scheme is to exploit an adversary to break the assumption (for instance, the partial-domain one-wayness of the permutation φ_{pk}). For that, we must simulate all the resources that the attacker can access, namely, the oracles $\mathcal{G}$, $\mathcal{H}$ but also the decryption oracle $\mathcal{D}_{\mathsf{sk}}$. For the above 2-round OAEP, the decryption oracle does not seem simulatable. The following attack game uses the same arguments as the counter-example shown by Shoup against the original OAEP security result [22]. Let us consider an attacker who chooses s, s' and calls for $\mathcal{H}$ to get respectively $h = \mathcal{H}(s)$ and $h' = \mathcal{H}(s')$. Then it chooses t and computes $c = \varphi_{\mathsf{pk}}(s, t)$. If it asks c to $\mathcal{D}_{\mathsf{sk}}$, it gets the corresponding plaintext m. Then, it computes $t' = t \oplus h \oplus h'$ and $c' = \varphi_{\mathsf{pk}}(s', t')$. If it asks c' to $\mathcal{D}_{\mathsf{sk}}$, it gets the corresponding plaintext m'. One can easily see that, since $r' = r$, the relation $m \oplus m = s \oplus s'$ should hold. But if the simulator can not detect that $r' = r$, it can not output a consistent value for m'.

Unfortunately, we did not find any easy way to make a consistent simulation for the 2-round OAEP. But a 3-round is more promising.

4.2 Description of the 3-round OAEP

The public key is any trapdoor (partial-domain) one-way bijection φ_{pk} from a set E to a set F, while the private key is the inverse ψ_{sk}. For the sake of generality, we do not stick to binary sets (of the form $\{0,1\}^k$): we just assume that there is an integer κ such that:

$$\{0\}^\kappa \times \{0,1\}^{k+\ell} \subseteq \mathrm{E} \subseteq \{0,1\}^{\kappa+k+\ell} \qquad \text{(identified to } \{0,1\}^\kappa \times \{0,1\}^k \times \{0,1\}^\ell\text{)}.$$

However, note that in the case that $\mathrm{E} \neq 0^\kappa \| \{0,1\}^{k+\ell}$ we won't get (as announced) a surjective encryption. But contrary to all the previous IND-CCA schemes, the proportion of valid ciphertexts (*i.e.*, which are reachable) is greater than $1/2^\kappa$, which is not negligible: for efficient applications with RSA, it can be equal to $1/2$, or even 1 (by loosing a factor 2 in efficiency, one can get $\kappa = 0$, with the iterated-RSA [2]).

The encryption and decryption algorithms use three hash functions: $\mathcal{F}$, $\mathcal{G}$, $\mathcal{H}$ (assumed to behave like random oracles in the security analysis):

$$\mathcal{F} : \{0,1\}^k \to \{0,1\}^\ell \qquad \mathcal{G} : \{0,1\}^\ell \to \{0,1\}^k \qquad \mathcal{H} : \{0,1\}^{k+\kappa} \to \{0,1\}^\ell.$$

Encryption Algorithm: The space of the plaintexts is $\mathcal{M} = \{0,1\}^\ell$, the encryption algorithm uses random coins in $\mathcal{R} = \{0,1\}^k$, and outputs a ciphertext c into F: on a plaintext $m \in \mathcal{M}$, and a random $r \in \mathcal{R}$, one computes

$$s = m \oplus \mathcal{F}(r) \qquad t = r \oplus \mathcal{G}(s) \qquad u = s \oplus \mathcal{H}(0^\kappa \| t) \qquad c = \varphi_{\mathsf{pk}}(0^\kappa, t, u).$$

Decryption Algorithm: On a ciphertext c, one first computes $(B, t, u) = \varphi_{\mathsf{sk}}(c)$, where $B \in \{0,1\}^\kappa$, $t \in \{0,1\}^k$, $u \in \{0,1\}^\ell$ and then

$$s = u \oplus \mathcal{H}(B \| t) \qquad r = t \oplus \mathcal{G}(s) \qquad m = s \oplus \mathcal{F}(r).$$

4.3 Security Result

About the 3-round OAEP, one can claim the following security result, which states that the IND-CCA security level is achieved under the (set) partial-domain one-wayness of the trapdoor permutation φ [15].

Theorem 2. *Let $\mathcal{A}$ be any chosen-ciphertext adversary against the 3-round OAEP construction with the trapdoor permutation family φ, within time τ. After q_f, q_g, q_h and q_d queries to the random oracles $\mathcal{F}$, $\mathcal{G}$ and $\mathcal{H}$, and the decryption oracle respectively,*

$$\begin{aligned}\mathsf{Adv}_{\pi}^{\mathsf{ind\text{-}cca}}(\tau) \leq 2^{\kappa} \times \mathsf{Succ}_{\varphi}^{\mathsf{s\text{-}pd\text{-}ow}}(\tau + q_g \cdot q_h \times T_{\varphi} + q_d \cdot T_{lu}, q_h)\\ \frac{q_f}{2^k} + \frac{q_g}{2^\ell} + 2^{\kappa} \times \left(\frac{q_d(2q_g + q_d)}{2^\ell} + \frac{q_d(3q_f + 2q_d)}{2^k} \right)\end{aligned}$$

where T_{φ} is the time complexity for evaluating φ, and T_{lu} is the time complexity for a look up in a list.

Let us recall the definition of the (set) partial-domain one-wayness in our particular case, where $\mathcal{A}$ is any algorithm which outputs a subset of $\{0,1\}^k$ of size q:

$$\mathsf{Succ}_{\varphi}^{\mathsf{s\text{-}pd\text{-}ow}}(\mathcal{A}, q) = \Pr_{\substack{\mathsf{pk}, \\ (B,t,u)\in E}}[t \in \mathcal{A}(\varphi_{\mathsf{pk}}(B,t,u))] \text{ and } \mathsf{Succ}_{\varphi}^{\mathsf{s\text{-}pd\text{-}ow}}(\tau, q) = \max_{|\mathcal{A}|\leq\tau} \left\{ \mathsf{Succ}_{\varphi}^{\mathsf{s\text{-}pd\text{-}ow}}(\mathcal{A}, q) \right\},$$

is small for any reasonable time bound τ.

4.4 Sketch of the Proof

The goal of the proof is again to simulate the oracles. For simulating the random oracles, we use lists as usual to store the known answers. We simulate the decryption oracle as follows: when we receive a query y, either the corresponding s and t have both been asked to $\mathcal{G}$ and $\mathcal{H}$, we can extract m, or one of them has not been asked, we can safely answer a random plaintext. However, such a plaintext-ciphertext relation implicitly defines several relations about the random oracles $\mathcal{F}$, $\mathcal{G}$ and $\mathcal{H}$. We show that it is still possible to answer consistently. The challenge ciphertext also implicitly defines relations. We show that possible inconsistencies with the latter relations can not be detected by the adversary unless it has partially inverted the function φ_{pk} on the challenge ciphertext.

The proof is provided by a sequence of games, but for clarity reasons, we briefly explain only the distances between two consecutive games. The formal and full proofs are provided in the Appendix B.

Game $\mathbf{G}_0$: The adversary is fed with the public key pk, and outputs a pair of messages (m_0, m_1). Next a challenge ciphertext is produced by flipping a coin b and producing a ciphertext $c^\star$ of $m^\star = m_b$. This ciphertext comes from a random $r^\star \leftarrow \{0,1\}^k$ and $c^\star = \mathcal{E}(m_b, r^\star) = \varphi_{\mathsf{pk}}(0^\kappa, t, u)$. On input $c^\star$, A_2 outputs bit b' in the time t. We denote by S_0 the event $b' = b$ and use the same notation S_n in any game $\mathbf{G}_n$ below. Note that the adversary is given access to the decryption oracle $\mathcal{D}_{\mathsf{sk}}$ during both steps of the attack. The adversary can also ask the random oracles $\mathcal{F}$, $\mathcal{G}$, and $\mathcal{H}$.

Game $\mathbf{G}_1$: The simulation in this game is presented on the Figure 2. We simulate the way that the challenge $c^\star$ is generated as the challenger would do, and we simulate the random oracles $\mathcal{F}$, $\mathcal{G}$, and $\mathcal{H}$, as well as the decryption oracle $\mathcal{D}_{\mathsf{sk}}$, by maintaining lists $\mathcal{F}$-List, $\mathcal{G}$-List,$\mathcal{H}$-List and $\mathcal{D}$-List to deal with identical queries, since they all are deterministic. Since the simulation is perfect, we directly derive that

$$\Pr[\mathsf{S}_1] = \Pr[\mathsf{S}_0]. \tag{1}$$

Game $\mathbf{G}_2$: We manufacture the challenge $c^\star$ independently of anything else.

▶**Rule** $\mathsf{Chal}^{(2)}$

| Choose randomly ahead of time $c^+ \xleftarrow{R} \mathrm{F}$ and set $c^\star = c^+$.

Lemma 3. *Let us note (B^+, t^+, u^+) the pre-image of the challenge c^+. We denote by AskH_2 the event that $B^+ \| t^+$ has been asked to $\mathcal{H}$. Then,*

$$\Pr[\mathsf{S}_1] \leq \frac{1}{2} + \frac{q_f}{2^k} + \frac{q_g}{2^\ell} + 2^{\kappa} \times \Pr[\mathsf{AskH}_2]. \tag{2}$$

Proof (Full proof in the Appendix B.1). The main idea in simulating this game is that we make the components of the challenge $c^\star$ (namely $r^\star$, $f^\star$, $s^\star$, $g^\star$, $t^\star$, $h^\star$, $u^\star$ and $c^\star$) independent to $m^\star$. We can do this by choosing ahead of time random values for $r^\star$, $s^\star$, and $t^\star$, and we can see that a difference occurs when one of these values is asked to the corresponding oracle. On the other hand, when the challenge is independent to $m^\star$, the attacker has only the chance of one half to guess the bit b. □

Game G$_3$: In this game, we modify the simulation of the decryption oracle, by outputting a random message when the ciphertext has not been "correctly" encrypted. We thereafter define in a consistent way the values of the random oracles:

▶**Rule** Decrypt-TnoS$^{(3)}$

```
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$ and $g \overset{R}{\leftarrow} \{0,1\}^k$,
    then define $r = t \oplus g$ and $f = m \oplus s$.
Add $(r, f)$ in $\mathcal{F}$-List, and $(s, g)$ in $\mathcal{G}$-List.
```

▶**Rule** Decrypt-noT$^{(3)}$

```
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$, $h \overset{R}{\leftarrow} \{0,1\}^\ell$ and $g \overset{R}{\leftarrow} \{0,1\}^k$,
    then define $s = u \oplus h$, $r = t \oplus g$ and $f = m \oplus s$.
Add $(r, f)$ in $\mathcal{F}$-List, $(s, g)$ in $\mathcal{G}$-List, and $(B, t, h)$ in $\mathcal{H}$-List.
```

Lemma 4.

$$|\Pr[\mathsf{AskH}_3] - \Pr[\mathsf{AskH}_2]| \leq \frac{q_d(q_g + q_d)}{2^\ell} + 2\frac{q_d(q_f + q_d)}{2^k}. \tag{3}$$

Proof (Full proof in the Appendix B.2). In the proof, one successively modifies the simulation of the decryption oracle, just changing the order of elements to be randomly chosen, so that the decryption of a ciphertext which has not been correctly encrypted is a truly random plaintext. □

Game G$_4$: In this game, we delay the explicit definitions of some oracle answers implicitly defined by some plaintext-ciphertext relations: we do not introduce them during the simulation of the decryption oracle, but when s is asked to $\mathcal{G}$. Some problems may appear if the implicitly defined answers are asked before $\mathcal{G}(s)$ is queried.

▶**Rule** Decrypt-TnoS$^{(4)}$

```
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$.
```

▶**Rule** Decrypt-noT$^{(4)}$

```
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$.
```

▶**Rule** EvalGAdd$^{(4)}$

```
Look up for $(B, t, h) \in \mathcal{H}$-List and $(m, c) \in \mathcal{D}$-List such that $c = \varphi_{\mathsf{pk}}(B, t, h \oplus s)$.
If the record is found, we compute $r = t \oplus g$ and $f = m \oplus s$, and finally add $(r, f)$
in $\mathcal{F}$-List.
```

Lemma 5.

$$|\Pr[\mathsf{AskH}_4] - \Pr[\mathsf{AskH}_3]| \leq \frac{q_d \cdot q_f}{2^k} + \frac{q_d \cdot q_g}{2^\ell}. \tag{4}$$

Proof (Full proof in the Appendix B.3). Since we don't store anymore (r, f), (s, g), (B, t, h), inconsistencies could occur when $B\|t$, s or r are asked. For solving this problem, we modify the rule EvalGAdd by defining in a consistent way $\mathcal{F}(r)$ at the moment that s is asked to $\mathcal{G}$. But there is still a problem if r is asked before $\mathcal{G}(s)$ is queried, or if s is asked before $\mathcal{H}(B\|t)$ is queried. □

Game $\mathbf{G}_5$: We now complete the simulation of the oracle $\mathcal{D}_{\mathsf{sk}}$. We don't ask any query to ψ_{sk}. Intuitively, if both $B\|t$ and s have been asked, we can easily find them, and then m. Otherwise, we give a random answer as in the game $\mathbf{G}_4$.

▶**Rule** Decrypt-Init$^{(5)}$

Look up for $(B,t,h) \in \mathcal{H}$-List and $(s,g) \in \mathcal{G}$-List such that $\varphi_{\mathsf{pk}}(B,t,s \oplus h) = c$.

- if the record is found, we furthermore define $u = s \oplus h$.
- otherwise, we take $B = \perp, t = \perp, u = \perp$.

▶**Rule** Decrypt-TS$^{(5)}$

$r = t \oplus g, \quad f = \mathcal{F}(r), \quad m = s \oplus f.$

The two games $\mathbf{G}_5$ and $\mathbf{G}_4$ are perfectly indistinguishable. In fact, in the first case, nothing is modified and in the second case, by making $B = \perp, t = \perp, u = \perp$, the answer of the decryption oracle for the question c will be a random m as in the game $\mathbf{G}_4$:

$$\Pr[\mathsf{AskH}_5] = \Pr[\mathsf{AskH}_4]. \tag{5}$$

Simply outputting the list of queries to $\mathcal{H}$ during this game, one gets:

$$\Pr[\mathsf{AskH}_5] \leq \mathsf{Succ}_{\varphi}^{\mathsf{s\text{-}pd\text{-}ow}}(\tau', q_h), \tag{6}$$

where τ' is the running time of the simulation in this game: $\tau' \leq q_g \cdot q_h \times T_\varphi + q_d \times T_{lu}$. We can indeed perform the simulation granted an additional list $\mathcal{GH}$-List which contains all the tuples (B,t,h,s,g,y) where $(B,t,h) \in \mathcal{H}$-List, $(s,g) \in \mathcal{G}$-List and $y = \varphi_{\mathsf{pk}}(B,t,s \oplus h)$. This concludes the proof of the Theorem.

4.5 Special Cases

In the particular but classical case where $\kappa = 0$ and $k \leq \ell$, one can claim

Theorem 6. *Let $\mathcal{A}$ be any chosen-ciphertext adversary against the 3-round OAEP construction with the trapdoor permutation family φ, within time τ. After q_o and q_d queries to the random oracles and the decryption oracle respectively,*

$$\mathsf{Adv}_{\pi}^{\mathsf{ind\text{-}cca}}(\tau) \leq \mathsf{Succ}_{\varphi}^{\mathsf{s\text{-}pd\text{-}ow}}(\tau + q_o^2 \times T_\varphi + q_d \times T_{lu}, q_o) + \frac{2q_o + q_d(5q_o + 2q_d)}{2^k}$$

where T_φ is the time complexity for evaluating φ, and T_{lu} is the time complexity for a look up in a list.

5 Conclusion

We have described the Full-Domain Permutation encryption which is IND-CCA without redundancy and provides an optimal bandwidth. In the random oracle model, we have shown that the absence redundancy can be obtained by considering the 3-round OAEP construction. However, the bandwidth is not optimal, and the security relies on the strong partial-domain one-wayness assumption.

Acknowledgments

We thank Anand Desai for fruitful discussions.

References

1. M. Abdalla, M. Bellare, and P. Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *CT – RSA '01*, LNCS 2020, pages 143–158. Springer-Verlag, Berlin, 2001.
2. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-Privacy in Public-Key Encryption. In *Asiacrypt '01*, LNCS 2248, pages 566–582. Springer-Verlag, Berlin, 2001.
3. 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.
4. 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.
5. 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.
6. M. Bellare and P. Rogaway. The Exact Security of Digital Signatures – How to Sign with RSA and Rabin. In *Eurocrypt '96*, LNCS 1070, pages 399–416. Springer-Verlag, Berlin, 1996.
7. M. Bellare and A. Sahai. Non-Malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization. In *Crypto '99*, LNCS 1666, pages 519–536. Springer-Verlag, Berlin, 1999.
8. D. Boneh. Simplified OAEP for the RSA and Rabin Functions. In *Crypto '01*, LNCS 2139, pages 275–291. Springer-Verlag, Berlin, 2001.
9. J.-S. Coron. On the Exact Security of Full-Domain-Hash. In *Crypto '00*, LNCS 1880, pages 229–235. Springer-Verlag, Berlin, 2000.
10. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In *Crypto '98*, LNCS 1462, pages 13–25. Springer-Verlag, Berlin, 1998.
11. A. Desai. New Paradigms for Constructing Symmetric Encryption Schemes Secure Against Chosen-Ciphertext Attack. In *Crypto '00*, LNCS 1880, pages 394–412. Springer-Verlag, Berlin, 2000.
12. D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. In *Proc. of the 23rd STOC*. ACM Press, New York, 1991.
13. E. Fujisaki and T. Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. In *PKC '99*, LNCS 1560, pages 53–68. Springer-Verlag, Berlin, 1999.
14. E. Fujisaki and T. Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In *Crypto '99*, LNCS 1666, pages 537–554. Springer-Verlag, Berlin, 1999.
15. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA–OAEP is Secure under the RSA Assumption. *Journal of Cryptology*, 2003. To appear.
16. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
17. M. Naor and M. Yung. Public-Key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *Proc. of the 22nd STOC*, pages 427–437. ACM Press, New York, 1990.
18. T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In *CT – RSA '01*, LNCS 2020, pages 159–175. Springer-Verlag, Berlin, 2001.
19. D. Pointcheval. Chosen-Ciphertext Security for any One-Way Cryptosystem. In *PKC '00*, LNCS 1751, pages 129–146. Springer-Verlag, Berlin, 2000.
20. 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.
21. R. Rivest, A. Shamir, and A. Tauman, How to Leak a Secret. In *Asiacrypt '01*, LNCS 2248, pages 552–565. Springer-Verlag, Berlin, 2001.
22. V. Shoup. OAEP Reconsidered. In *Crypto '01*, LNCS 2139, pages 239–259. Springer-Verlag, Berlin, 2001.

A Security Proof of the φ-FDP Encryption

Game $\mathbf{G}_0$: This is the attack game, in the random permutation model. Several oracles are thus available to the adversary: two random permutation oracles ($\mathcal{P}$ and $\mathcal{P}^{-1}$) and the decryption oracle $\mathcal{D}_{\mathsf{sk}}$. The adversary $\mathcal{A} = (A_1, A_2)$ runs its attack in two steps. First, A_1 is fed with the public key pk, and outputs a pair of messages (m_0, m_1). Next a challenge ciphertext is produced by the challenger, which flips a coin b and computes a ciphertext $c^\star$ of $m^\star = m_b$. This ciphertext comes from a random $r^\star \overset{R}{\leftarrow} \{0,1\}^k$ and

$$c^\star = \mathcal{E}(m_b, r^\star) = \varphi_{\mathsf{pk}}(\mathcal{P}(m_b, r^\star)).$$

In the second step, on input $c^\star$, A_2 outputs a bit b'. We denote by S_0 the event $b' = b$ and use the same notation S_n in any game $\mathbf{G}_n$. Note that the adversary is given access to the decryption oracle $\mathcal{D}_{\mathsf{sk}}$ during both steps of the attack.

By definition,

$$\varepsilon = \mathsf{Adv}_{\pi}^{\mathsf{ind\text{-}cca}}(\mathcal{A}) = 2\Pr[\mathsf{S}_0] - 1. \tag{7}$$

In the games below, we furthermore assume that when the game aborts or stops with no answer b' outputted by the adversary $\mathcal{A}$, we choose this bit b' at random, which in turn defines the actual value of the event S_n.

Game $\mathbf{G}_1$: A perfect simulation of the real attack game is described on the Figure 1. Actually, the rule $\mathsf{Chal}^{(1)}$ simulates the way the challenge $c^\star$ is generated, exactly as the challenger would do. Besides, we simulate the two random permutation oracles $\mathcal{P}$ and $\mathcal{P}^{-1}$, and the decryption oracle $\mathcal{D}_{\mathsf{sk}}$, by maintaining a list $\mathcal{P}$-List, and using a truly random permutation P and its inverse P^{-1}.

Note that when we "store" an element (m, r, p, y) in $\mathcal{P}$-List: if $p \neq \perp$, and if there is already an element $(m, r, \star, y)$ in $\mathcal{P}$-List, we replace the latter by the former; if $p \neq \perp$, and if there is already an element (m, r, c, y). we don't append the element; otherwise, the tuple is appended to the list.

We see that the simulation is perfect since $\mathcal{P}$-List is not used. The latter list is only introduced for later simulation in the proof.

$$\Pr[\mathsf{S}_1] = \Pr[\mathsf{S}_0]. \tag{8}$$

Game $\mathbf{G}_2$: In this game, we modify the simulation of the oracle $\mathcal{P}$, so that the random permutation P is called at most once for each input. We use the following rule:

```
▶Rule EvalP$^{(2)}$
  | Look up for $(m, r, \alpha, y)$ in $\mathcal{P}$-List:
  |   – if the record is found,
  |       • if $\alpha \neq \perp$, $p = \alpha$;
  |       • otherwise, $p \overset{R}{\leftarrow} \{0,1\}^{\ell+k}$.
  |   – otherwise, $p = P(m, r)$.
```

Note that if $\alpha = \perp$, we likely give a wrong answer (the correct answer should be $\psi_{\mathsf{sk}}(y)$), we define this event BadP_2. Unless this event happens, the two games $\mathbf{G}_2$ and $\mathbf{G}_1$ are perfectly indistinguishable:

$$|\Pr[\mathsf{S}_2] - \Pr[\mathsf{S}_1]| \leq \Pr[\mathsf{BadP}_2], \tag{9}$$

Game $\mathbf{G}_3$: We now modify the same way the simulation of the oracle $\mathcal{P}^{-1}$ by using the following rule:

```
▶Rule InvP$^{(3)}$
  | Compute $c = \varphi_{\mathsf{pk}}(p)$ and look up for $(m, r, \alpha, c)$ in $\mathcal{P}$-List:
  |   – if the record is found, $(m, r)$ is defined,
  |   – otherwise we compute $(m, r) = P^{-1}(p)$.
```

The two games $\mathbf{G}_3$ and $\mathbf{G}_2$ are perfectly indistinguishable:

$$\Pr[\mathsf{S}_3] = \Pr[\mathsf{S}_2] \qquad \Pr[\mathsf{BadP}_3] = \Pr[\mathsf{BadP}_2]. \tag{10}$$

Game $\mathbf{G}_4$: In this game, we continue to modify the simulation of the oracles $\mathcal{P}$ and $\mathcal{P}^{-1}$, without asking any query at all to the random permutation P (and its inverse P^{-1}), but answering a random value for any new query: for a new (m, r), we answer $\mathcal{P}(m, r)$ by a random p, and for a new p we answer $\mathcal{P}^{-1}(p)$ by a random pair (m, r). We rewrite the rules as follows:

```
▶Rule EvalP$^{(4)}$
  | Look up for $(m, r, \alpha, y)$ in $\mathcal{P}$-List:
  |   – if the record is found,
  |       • if $\alpha \neq \perp$, $p = \alpha$;
  |       • otherwise, $p \overset{R}{\leftarrow} \{0,1\}^{\ell+k}$.
  |   – otherwise, $p \overset{R}{\leftarrow} \{0,1\}^{\ell+k}$.
```

▶**Rule** $\mathsf{InvP}^{(4)}$

> Compute $c = \varphi_{\mathsf{pk}}(p)$ and look up for (m, r, α, c) in $\mathcal{P}$-List:
>
> – if the record is found, (m, r) is defined,
> – otherwise we randomly choose (m, r) in $\{0,1\}^{\ell+k}$.

The two games $\mathbf{G}_4$ and $\mathbf{G}_3$ are perfectly indistinguishable unless a collision appears over (m, r) or p in $\mathcal{P}$-List, we define this event CollP_4. So, we have:

$$|\Pr[\mathsf{S}_4] - \Pr[\mathsf{S}_3]| \leq \Pr[\mathsf{CollP}_4] \qquad |\Pr[\mathsf{BadP}_4] - \Pr[\mathsf{BadP}_3]| \leq \Pr[\mathsf{CollP}_4]. \tag{11}$$

Since there are at most $q_d + q_p + 1$ elements in the $\mathcal{P}$-List, such a collision appears with probability bounded by $(q_p + q_d + 1)^2/2^{k+\ell+1}$:

$$\Pr[\mathsf{CollP}_4] \leq \frac{(q_p + q_d + 1)^2}{2^{k+\ell+1}}. \tag{12}$$

Game $\mathbf{G}_5$: Finally, we can simulate the decryption oracle $\mathcal{D}_{\mathsf{sk}}$ without inverting φ_{sk}:

▶**Rule** $\mathsf{Decrypt}^{(5)}$

> Look up for (m, r, α, c) in $\mathcal{P}$-List:
>
> 1. if the record is found, (m, r) is defined,
> 2. otherwise we randomly choose (m, r) in $\{0,1\}^{\ell+k}$.

The two games $\mathbf{G}_5$ and $\mathbf{G}_4$ are perfectly indistinguishable since in case 2, we do exactly as $\mathcal{P}^{-1}$ was simulated by the rule $\mathsf{InvP}^{(4)}$ in the previous game:

$$\Pr[\mathsf{S}_5] = \Pr[\mathsf{S}_4] \qquad \Pr[\mathsf{BadP}_5] = \Pr[\mathsf{BadP}_4]. \tag{13}$$

Game $\mathbf{G}_6$: We consider the elements of the form $(m, r, \perp, y)$ in $\mathcal{P}$-List (there are at most $q_d + 1$ elements). If there is a collision on m, we abort the game, which event is named CollM_6.

$$|\Pr[\mathsf{S}_6] - \Pr[\mathsf{S}_5]| \leq \Pr[\mathsf{CollM}_6] \qquad |\Pr[\mathsf{BadP}_6] - \Pr[\mathsf{BadP}_5]| \leq \Pr[\mathsf{CollM}_6]. \tag{14}$$

We see that for any $(m, r, \perp, y)$ except if $m = m^\star$, m is chosen randomly. As a consequence, a collision over m can be found with probability bounded by $(q_d + 1)^2/2^{\ell+1}$. Therefore,

$$\Pr[\mathsf{CollM}_6] \leq \frac{(q_d + 1)^2}{2^{\ell+1}}. \tag{15}$$

When collisions are excluded, for any m, there is at most one r (and a y) such that $(m, r, \perp, y) \in \mathcal{P}$-List. Therefore one can see that

$$\Pr[\mathsf{BadP}_6] \leq \frac{q_p}{2^k}. \tag{16}$$

Game $\mathbf{G}_7$: Now we suppress the element $(m^\star, r^\star, \perp, c^\star)$ from $\mathcal{P}$-List.

▶**Rule** $\mathsf{ChalAdd}^{(7)}$

> Do nothing.

The two games $\mathbf{G}_7$ and $\mathbf{G}_6$ are perfectly indistinguishable unless $(m^\star, r^\star)$ is asked for $\mathcal{P}$ (which event is included in event BadP_7, already excluded) or $p^\star$ is asked to $\mathcal{P}^{-1}$. We define the latter event $\mathsf{AskInvP}_7$. We have:

$$|\Pr[\mathsf{S}_7] - \Pr[\mathsf{S}_6]| \leq \Pr[\mathsf{AskInvP}_7]. \tag{17}$$

Since $(m^\star, r^\star, p^\star, c^\star)$ does not appear in $\mathcal{P}$-List, the adversary receives answers which are perfectly independent of the latter, and therefore, it has no advantage for guessing b:

$$\Pr[\mathsf{S}_7] = \frac{1}{2}. \tag{18}$$

Game $\mathbf{G}_8$: Instead of choosing $c^\star = \varphi_{\mathsf{pk}}(p^\star)$, we choose $c^\star$, uniformly at random.

▶**Rule** $\mathsf{Chal}^{(8)}$

$y \overset{R}{\leftarrow} \{0,1\}^{k+\ell}$; $c^\star = y$.

So, one implicitly defines $p^\star = \psi_{\mathsf{sk}}(y)$. Since the tuple $(m^\star, r^\star, p^\star, c^\star)$ is not used anywhere in the simulation, the two games $\mathbf{G}_8$ and $\mathbf{G}_7$ are perfectly indistinguishable:

$$\Pr[\mathsf{AskInvP}_8] = \Pr[\mathsf{AskInvP}_7]. \tag{19}$$

Finally, it is clear that when the event $\mathsf{AskInvP}_8$ happens, one can easily invert φ on y: with a look up into $\mathcal{P}$-List, on can extract p such that $y = \varphi_{\mathsf{pk}}(p)$. Therefore,

$$\Pr[\mathsf{AskInvP}_8] \leq \mathsf{Succ}_{\varphi}^{\mathsf{ow}}(\tau' + q_p \times T_\varphi), \tag{20}$$

where T_φ is the time for evaluating φ, and $\tau' \leq \tau + q_p \times T_\varphi$ is the running time of the simulation in the current game.

Conclusion. Finally, combining all the above equations:

$$\begin{aligned}\frac{\varepsilon}{2} &= \frac{1+\epsilon}{2} - \frac{1}{2} = \Pr[\mathsf{S}_0] - \Pr[\mathsf{S}_7] \\ &\leq \Pr[\mathsf{S}_0] - \Pr[\mathsf{S}_1] + \Pr[\mathsf{AskInvP}_7] + \Pr[\mathsf{CollM}_6] + \Pr[\mathsf{CollP}_4] + \Pr[\mathsf{BadP}_2] \\ &\leq \Pr[\mathsf{AskInvP}_8] + 2\Pr[\mathsf{CollM}_6] + 2\Pr[\mathsf{CollP}_4] + \Pr[\mathsf{BadP}_6].\end{aligned}$$

The equations (20), (15), (12) and (16) lead to

$$\frac{\varepsilon}{2} \leq \mathsf{Succ}_{\varphi}^{\mathsf{ow}}(\tau + 2q_p \times T_\varphi) + \frac{(q_d+1)^2}{2^\ell} + \frac{(q_p+q_d+1)^2}{2^{k+\ell}} + \frac{q_p}{2^k}.$$

B Complements for the Proof of the Theorem 2

B.1 Proof of Lemma 3

Game $\mathbf{G}_{1.1}$: For proving this lemma, we present a more detailed sequence of games from the game $\mathbf{G}_1$ to the game $\mathbf{G}_{1.2}$. We first make the value of the random seed $r^\star$ explicit and move its generation up-front.

▶**Rule** $\mathsf{Chal}^{(1.1)}$

The two values $r^+ \overset{R}{\leftarrow} \{0,1\}^k$, $f^+ \overset{R}{\leftarrow} \{0,1\}^\ell$ have been chosen ahead of time, then

$$r^\star = r^+, \quad f^\star = f^+, \quad s^\star = m^\star \oplus f^+, \quad g^\star = \mathcal{G}(s^\star),$$

$$t^\star = r^+ \oplus g^\star, \quad h^\star = \mathcal{H}(0^\kappa \| t^\star), \quad u^\star = s^\star \oplus h^\star.$$

Compute $c^\star = \varphi_{\mathsf{pk}}(0^\kappa, t^\star, u^\star)$.

The two games $\mathbf{G}_{1.1}$ and $\mathbf{G}_1$ are perfectly indistinguishable unless $r^\star$ has been asked for $\mathcal{F}$. We define this event $\mathsf{AskF}_{1.1}$. We have:

$$|\Pr[\mathsf{S}_{1.1}] - \Pr[\mathsf{S}_1]| \leq \Pr[\mathsf{AskF}_{1.1}]. \tag{21}$$

In this game, f^+ is used in (s,t) but does not appear in the computation since $\mathcal{F}(r^+)$ is not defined to be equal to f^+. Thus, the input to $\mathcal{A}_2$ follows a distribution that does not depend on b. Accordingly:

$$\Pr[\mathsf{S}_{1.1}] = \frac{1}{2}. \tag{22}$$

Game $\mathbf{G}_{1.2}$: In this game, instead of defining $s^\star$ from $f^\star$ which is a random value f^+, we randomly choose $s^\star$ and then we define f^+ from $s^\star$. Because $s^\star$ is chosen randomly, we give a random answer for the question $s^\star$ to $\mathcal{G}$.

▶**Rule** $\mathsf{Chal}^{(1.2)}$

The values $r^+ \overset{R}{\leftarrow} \{0,1\}^k$, $s^+ \overset{R}{\leftarrow} \{0,1\}^\ell$, $g^+ \overset{R}{\leftarrow} \{0,1\}^k$ have been chosen ahead of time, then

$$r^\star = r^+ \quad s^\star = s^+ \quad g^\star = g^+ \quad f^\star = s^+ \oplus m^\star$$

$$t^\star = r^+ \oplus g^+ \quad h^\star = \mathcal{H}(0^\kappa \| t^\star) \quad u^\star = s^+ \oplus h^\star.$$

Compute $c^\star = \varphi_{\mathsf{pk}}(0^\kappa, t^\star, u^\star)$.

The two games $\mathbf{G}_{1.2}$ and $\mathbf{G}_{1.1}$ are perfectly indistinguishable unless $s^\star$ is asked for $\mathcal{G}$. We define this event $\mathsf{AskG}_{1.2}$. We have:

$$|\Pr[\mathsf{AskF}_{1.2}] - \Pr[\mathsf{AskF}_{1.1}]| \leq \Pr[\mathsf{AskG}_{1.2}]. \tag{23}$$

In this game, $r^+ = t^\star \oplus g^+$ is uniformly distributed, and independently of the adversary's view since g^+ is never revealed:

$$\Pr[\mathsf{AskF}_{1.2}] = \frac{q_f}{2^k}. \tag{24}$$

Game $\mathbf{G}_{1.3}$: Similarly to the above game, instead of defining $t^\star$ from a random g^+, we randomly choose $t^\star$ and then we define g^+ from $t^\star$. Because $t^\star$ is chosen randomly, we give a random answer for the question $(0^\kappa \| t^\star)$ to $\mathcal{H}$.

▶**Rule** $\mathsf{Chal}^{(1.3)}$

The values $r^+ \overset{R}{\leftarrow} \{0,1\}^k$, $s^+ \overset{R}{\leftarrow} \{0,1\}^\ell$, $t^+ \overset{R}{\leftarrow} \{0,1\}^k$, $h^+ \overset{R}{\leftarrow} \{0,1\}^\ell$ have been chosen ahead of time, then

$$r^\star = r^+ \quad s^\star = s^+ \quad t^\star = t^+ \quad h^\star = h^+$$

$$f^\star = s^+ \oplus m^\star \quad g^\star = t^+ \oplus r^+ \quad u^\star = s^+ \oplus h^+.$$

Compute $c^\star = \varphi_{\mathsf{pk}}(0^\kappa, t^\star, u^\star)$.

The two games $\mathbf{G}_{1.3}$ and $\mathbf{G}_{1.2}$ are perfectly indistinguishable unless $0^\kappa \| t^\star$ is asked for $\mathcal{H}$. We define this event $\mathsf{AskH}_{1.3}$. We have:

$$|\Pr[\mathsf{AskG}_{1.3}] - \Pr[\mathsf{AskG}_{1.2}]| \leq \Pr[\mathsf{AskH}_{1.3}]. \tag{25}$$

In this game, $s^+ = u^\star \oplus h^+$ is uniformly distributed, and independently of the adversary's view since h^+ is never revealed:

$$\Pr[\mathsf{AskG}_{1.3}] = \frac{q_g}{2^\ell}.$$

Game $\mathbf{G}_{1.4}$: We manufacture the challenge $c^\star$ independently of anything else.

▶**Rule** $\mathsf{Chal}^{(1.4)}$

The values $t^+ \overset{R}{\leftarrow} \{0,1\}^k$, $u^+ \overset{R}{\leftarrow} \{0,1\}^\ell$ have been chosen ahead of time.
Compute $c^\star = \varphi_{\mathsf{pk}}(0^\kappa, t^+, u^+)$.

The distribution of $c^\star$ remains the same:

$$\Pr[\mathsf{AskH}_{1.4}] = \Pr[\mathsf{AskH}_{1.3}]. \tag{26}$$

Game $\mathbf{G}_{1.5}$: We choose the challenge $c^\star$ uniformly in the space F.

▶**Rule** $\mathsf{ChalC}^{(1.5)}$

The value $c^+ \overset{R}{\leftarrow} \mathrm{F}$ is chosen randomly ahead of time, then $c^\star = c^+$.

We can write c^+ as $\varphi_{\mathsf{pk}}(B^+, t^+, h^+)$. We define $\mathsf{AskH}_{1.5}$ the event that $B^+\|t^+$ is asked to $\mathcal{H}$. In the case $B^+ = 0^\kappa$, which event is denoted by GoodB and which probability is at least $1/2^\kappa$, this game is identical to the previous one:

$$\begin{aligned}\Pr[\mathsf{AskH}_{1.5}] &= \Pr[\mathsf{AskH}_{1.5} \wedge \mathsf{GoodB}] + \Pr[\mathsf{AskH}_{1.5} \wedge \neg\mathsf{GoodB}] \\ &\geq \Pr[\mathsf{AskH}_{1.5}|\mathsf{GoodB}] \cdot \Pr[\mathsf{GoodB}] \geq \Pr[\mathsf{AskH}_{1.4}] \cdot \frac{1}{2^\kappa}.\end{aligned} \quad (27)$$

To conclude the proof of the lemma, one first notes that the games $\mathbf{G}_{1.5}$ and $\mathbf{G}_2$ are identical, and thus $\Pr[\mathsf{AskH}_{1.5}] = \Pr[\mathsf{AskH}_2]$. Then, combining all the above equations, on gets

$$\begin{aligned}\Pr[\mathsf{S}_1] &\leq \Pr[\mathsf{S}_{1.1}] + \Pr[\mathsf{AskF}_{1.1}] \leq \frac{1}{2} + \Pr[\mathsf{AskF}_{1.1}] \\ &\leq \frac{1}{2} + \Pr[\mathsf{AskF}_{1.2}] + \Pr[\mathsf{AskG}_{1.2}] \\ &\leq \frac{1}{2} + \Pr[\mathsf{AskF}_{1.2}] + \Pr[\mathsf{AskG}_{1.3}] + 2^\kappa \cdot \Pr[\mathsf{AskH}_{1.5}] \\ &\leq \frac{1}{2} + \frac{q_f}{2^k} + \frac{q_g}{2^\ell} + 2^\kappa \cdot \Pr[\mathsf{AskH}_2].\end{aligned}$$

B.2 Proof of Lemma 4

Game $\mathbf{G}_{2.1}$: First, we modify the rule Decrypt-noT by not calling anymore the oracles $\mathcal{G}$ and $\mathcal{H}$. Let us remind that the adversary asks a $\mathcal{D}$-query on $c = \varphi_{\mathsf{pk}}(B, t, u)$ such that $\mathcal{H}(B\|t)$ has never been queried.

▶**Rule** Decrypt-noT$^{(2.1)}$

```
Choose $h \overset{R}{\leftarrow} \{0,1\}^\ell$ and set $s = u \oplus h$.
Choose $g \overset{R}{\leftarrow} \{0,1\}^k$ and set $r = t \oplus g$.
Compute $f = \mathcal{F}(r)$ and set $m = s \oplus f$.
Add $(s, g)$ in $\mathcal{G}$-List, $(B, t, h)$ in $\mathcal{H}$-List.
```

The two games $\mathbf{G}_{2.1}$ and $\mathbf{G}_2$ are perfectly indistinguishable unless s is already in $\mathcal{G}$-List. Because $B\|t$ has not been queried to $\mathcal{H}$, $h = \mathcal{H}(B\|t)$ is uniformly distributed and therefore, we can consider s as a uniform variable. So, the probability that s has already been queried to $\mathcal{G}$ is $(q_g + q_d)/2^\ell$:

$$|\Pr[\mathsf{AskH}_{2.1}] - \Pr[\mathsf{AskH}_2]| \leq q_d(q_g + q_d)/2^\ell. \quad (28)$$

Game $\mathbf{G}_{2.2}$: In this game, we modify again the rule Decrypt-noT$^{(2.2)}$ by not querying the oracle $\mathcal{F}$ either:

▶**Rule** Decrypt-noT$^{(2.2)}$

```
Choose $h \overset{R}{\leftarrow} \{0,1\}^\ell$ and set $s = u \oplus h$.
Choose $g \overset{R}{\leftarrow} \{0,1\}^k$ and set $r = t \oplus g$.
Choose $f \overset{R}{\leftarrow} \{0,1\}^\ell$ and set $m = s \oplus f$.
Add $(r, f)$ in $\mathcal{F}$-List, $(s, g)$ in $\mathcal{G}$-List, $(B, t, h)$ in $\mathcal{H}$-List.
```

The two games $\mathbf{G}_{2.2}$ and $\mathbf{G}_{2.1}$ are perfectly indistinguishable unless r is already in $\mathcal{F}$-List. Since g is randomly chosen, we can consider r as a uniform variable. So, the probability that r has already been queried to $\mathcal{F}$ is less than $(q_f + q_d)/2^k$:

$$|\Pr[\mathsf{AskH}_{2.2}] - \Pr[\mathsf{AskH}_{2.1}]| \leq q_d(q_f + q_d)/2^k. \quad (29)$$

Game $\mathbf{G}_{2.3}$: Still about the rule Decrypt-noT, instead of defining m from a random f, we first choose m and then we define f from m:

▶**Rule** Decrypt-noT$^{(2.3)}$

```
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$.
Choose $h \overset{R}{\leftarrow} \{0,1\}^\ell$ and set $s = u \oplus h$.
Choose $g \overset{R}{\leftarrow} \{0,1\}^k$ and set $r = t \oplus g$.
Compute $f = m \oplus s$.
Add $(r, f)$ in $\mathcal{F}$-List, $(s, g)$ in $\mathcal{G}$-List, $(B, t, h)$ in $\mathcal{H}$-List.
```

The two games $\mathbf{G}_{2.3}$ and $\mathbf{G}_{2.2}$ are perfectly indistinguishable:

$$\Pr[\mathsf{AskH}_{2.3}] = \Pr[\mathsf{AskH}_{2.2}]. \tag{30}$$

Game $\mathbf{G}_{2.4}$: We now modify the rule Decrypt-TnoS by not calling anymore the oracles $\mathcal{F}$ and $\mathcal{G}$. About this rule, the adversary asks for the decryption of $c = \varphi_{\mathsf{pk}}(B, t, u)$ such that $h = \mathcal{H}(B\|t)$ is known, but $s = u \oplus h$ has never been queried to $\mathcal{G}$.

▶**Rule** Decrypt-TnoS$^{(2.4)}$

```
Choose $g \overset{R}{\leftarrow} \{0,1\}^k$ and set $r = t \oplus g$.
Choose $f \overset{R}{\leftarrow} \{0,1\}^\ell$ and set $m = s \oplus f$.
Add $(r, f)$ in $\mathcal{F}$-List, $(s, g)$ in $\mathcal{G}$-List.
```

The two games $\mathbf{G}_{2.4}$ and $\mathbf{G}_{2.3}$ are perfectly indistinguishable unless r is already in $\mathcal{F}$-List. Since g is randomly chosen (s is not in $\mathcal{G}$-List), we can consider r as a uniform variable. So, the probability that r is queried to $\mathcal{F}$ is less than $(q_f + q_d)/2^k$:

$$|\Pr[\mathsf{AskH}_{2.4}] - \Pr[\mathsf{AskH}_{2.3}]| \leq q_d(q_f + q_d)/2^k. \tag{31}$$

Game $\mathbf{G}_{2.5}$: As above, in the rule Decrypt-TnoS, instead of defining m from a random f, we first choose m and then we define f from m:

▶**Rule** Decrypt-TnoS$^{(2.5)}$

```
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$.
Choose $g \overset{R}{\leftarrow} \{0,1\}^k$ and set $r = t \oplus g$.
Compute $f = m \oplus s$.
Add $(r, f)$ in $\mathcal{F}$-List, $(s, g)$ in $\mathcal{G}$-List.
```

The two games $\mathbf{G}_{2.5}$ and $\mathbf{G}_{2.4}$ are perfectly indistinguishable:

$$\Pr[\mathsf{AskH}_{2.5}] = \Pr[\mathsf{AskH}_{2.4}]. \tag{32}$$

B.3 Proof of Lemma 5

Game $\mathbf{G}_{3.1}$: In this game, we don't store anymore (s, g) in $\mathcal{G}$-List, nor (r, f) in $\mathcal{F}$-List and we modify the simulation of $\mathcal{G}$, so that $\mathcal{F}$-List is built as soon as possible:

▶**Rule** Decrypt-TnoS$^{(3.1)}$

```
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$.
```

▶**Rule** Decrypt-noT$^{(3.1)}$

```
Choose $h \overset{R}{\leftarrow} \{0,1\}^\ell$.
Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$.
Add $(B, t, h)$ in $\mathcal{H}$-List.
```

▶**Rule** $\mathsf{EvalGAdd}^{(3.1)}$

Search $(B,t,h) \in \mathcal{H}$-List and $(m,c) \in \mathcal{D}$-List such that $c = \varphi_{\mathsf{pk}}(B,t,h \oplus s)$. If the record is found, we compute $r = t \oplus g$, $f = m \oplus s$ and add (r,f) in $\mathcal{F}$-List.

The two games $\mathbf{G}_{3.1}$ and $\mathbf{G}_3$ are perfectly indistinguishable unless r is asked to $\mathcal{F}$ before s is asked to $\mathcal{G}$, we denote this event by AskRbS, In fact, if r is asked after s, at the moment that s is asked, by the above simulation of $\mathcal{G}$, we will find out (B,t,h) and therefore (r,f) is computed and added in $\mathcal{F}$-List as in the game $\mathbf{G}_3$.

$$|\Pr[\mathsf{AskH}_{3.1}] - \Pr[\mathsf{AskH}_3]| \leq \Pr[\mathsf{AskRbS}_{3.1}]. \tag{33}$$

Until s is asked, g is a uniform variable, so is r. Therefore, the probability that r has been asked to $\mathcal{F}$ is $q_f/2^k$:

$$\Pr[\mathsf{AskRbS}_{3.1}] \leq q_d \cdot q_f/2^k. \tag{34}$$

Game $\mathbf{G}_{3.2}$: We continue to simulate the oracle $\mathcal{D}_{\mathsf{sk}}$. We use the following rule:

▶**Rule** $\mathsf{Decrypt\text{-}noT}^{(3.2)}$

Choose $m \overset{R}{\leftarrow} \{0,1\}^\ell$.

In this game, we don't store anymore (B,t,h) in $\mathcal{H}$-List. In the $\mathbf{G}_{3.1}$, for the question t, we answer randomly h, so the attacker in the two games $\mathbf{G}_{3.2}$ and $\mathbf{G}_{3.1}$ can not distinguish the answers of a question to $\mathcal{H}$. Nevertheless, $\mathcal{H}$-List has been changed and therefore, the answer for a question to $\mathcal{F}$ can be changed. We easily see that the two games $\mathbf{G}_{3.2}$ and $\mathbf{G}_{3.1}$ are perfectly indistinguishable unless s is asked to $\mathcal{G}$ before $B\|t$ is asked to $\mathcal{H}$, we denote this event by AskSbT, In fact, if s is asked to $\mathcal{G}$ after $B\|t$ is asked to $\mathcal{H}$, at the moment s is asked, by the above simulation of $\mathcal{G}$, we will find out (B,t,h) and therefore (r,f) is computed and added in $\mathcal{F}$-List as in the game $\mathbf{G}_{3.1}$.

$$|\Pr[\mathsf{AskH}_{3.2}] - \Pr[\mathsf{AskH}_{3.1}]| \leq \Pr[\mathsf{AskSbT}_{3.2}]. \tag{35}$$

Until $B\|t$ is asked to $\mathcal{H}$, h is a uniform variable, so is $s = u \oplus h$. Therefore, the probability that s has been asked to $\mathcal{G}$ is $q_g/2^\ell$:

$$\Pr[\mathsf{AskSbT}_{3.2}] \leq q_d \cdot q_g/2^\ell.$$

$\mathcal{P}$-Oracle

A query $\mathcal{P}(m, r)$ is answered by p, where

▶**Rule** EvalP$^{(1)}$
  | $p = P(m, r)$.

Furthermore, if (m, r) is a direct query from the adversary to $\mathcal{P}$, store the record $(m, r, p, \varphi_{\mathsf{pk}}(p))$ in $\mathcal{P}$-List.

$\mathcal{P}^{-1}$-Oracle

A query $\mathcal{P}^{-1}(p)$ is answered by (m, r), where

▶**Rule** InvP$^{(1)}$
  | $(m, r) = P^{-1}(p)$.

Furthermore, if p a direct query from the adversary to $\mathcal{P}^{-1}$, store the record $(m, r, p, \varphi_{\mathsf{pk}}(p))$ in $\mathcal{P}$-List.

$\mathcal{D}$-Oracle

A query $\mathcal{D}_{\mathsf{sk}}(c)$ is answered by m, where

▶**Rule** Decrypt$^{(1)}$
  | $p = \psi_{\mathsf{sk}}(c)$, and $(m, r) = \mathcal{P}^{-1}(p)$.

Store $(m, r, \perp, c)$ in $\mathcal{P}$-List.

Challenger

For two messages (m_0, m_1), flip a coin b and set $m^\star = m_b$, randomly choose $r^\star$.

▶**Rule** Chal$^{(1)}$
  | $p^\star = \mathcal{P}(m^\star, r^\star)$; $c^\star = \varphi_{\mathsf{pk}}(p^\star)$.

▶**Rule** ChalAdd$^{(1)}$
  | Add $(m^\star, r^\star, p^\star, c^\star)$ in $\mathcal{P}$-List.

Answer $c^\star$

Fig. 1. Formal Simulations of the IND-CCA Game against φ-FDP

$\mathcal{F}$, $\mathcal{G}$ and $\mathcal{H}$ Oracles

Query $\mathcal{F}(r)$: if a record (r, f) appears in $\mathcal{F}$-List, the answer is f.
Otherwise the answer f is chosen randomly: $f \in \{0,1\}^k$ and the record (r, f) is added in $\mathcal{F}$-List.

Query $\mathcal{G}(s)$: if a record (s, g) appears in $\mathcal{G}$-List, the answer is g.
Otherwise the answer g is chosen randomly: $g \in \{0,1\}^\ell$ and the record (s, g) is added in $\mathcal{G}$-List.

```
▶Rule EvalGAdd^(1)
    Do nothing
```

Query $\mathcal{H}(B\|t)$: if a record (B, t, h) appears in $\mathcal{H}$-List, the answer is h.
Otherwise the answer h is chosen randomly: $h \in \{0,1\}^k$ and the record (B, t, h) is added in $\mathcal{H}$-List.

$\mathcal{D}$ Oracle

Query $\mathcal{D}_{\mathsf{sk}}(c)$: if a record (m, c) appears in $\mathcal{D}$-List, the answer is m.
Otherwise the answer m is defined according to the following rules:

```
▶Rule Decrypt-Init^(1)
    Compute $(B, t, u) = \psi_{\mathsf{sk}}(c)$;
```

Look up for $(B, t, h) \in \mathcal{H}$-List:

```
– if the record is found, compute $s = u \oplus h$.
  Look up for $(s, g) \in \mathcal{G}$-List:
    • if the record is found
        ▶Rule Decrypt-TS^(1)
            $h = \mathcal{H}(B\|t)$,
            $s = u \oplus h$,    $g = \mathcal{G}(s)$,
            $r = t \oplus g$,    $f = \mathcal{F}(r)$,
            $m = s \oplus f$.
    • otherwise
        ▶Rule Decrypt-TnoS^(1)
            same as rule Decrypt-TS^(1).
– otherwise
    ▶Rule Decrypt-noT^(1)
        same as rule Decrypt-TS^(1).
```

Answer m and add (m, c) to $\mathcal{D}$-List.

Challenger

For two messages (m_0, m_1), flip a coin b and set $m^\star = m_b$, choose randomly $r^\star$, then answer $c^\star$, where

```
▶Rule Chal^(1)
    $f^\star = \mathcal{F}(r^\star)$,    $s^\star = m^\star \oplus f^\star$,
    $g^\star = \mathcal{G}(s^\star)$,    $t^\star = r^\star \oplus g^\star$,
    $h^\star = \mathcal{H}(0^\kappa\|t^\star)$,    $u^\star = s^\star \oplus h^\star$.
    Compute $c^\star = \varphi_{\mathsf{pk}}(0^\kappa, t^\star, u^\star)$.
```

Fig. 2. Formal Simulation of the IND-CCA Game against 3-OAEP