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 $m^\|0^k$, 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 IND-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 IND-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 IND-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 \( \pi \) 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 \( (pk, sk) \) of matching public and private keys.
- **The encryption algorithm** \( \mathcal{E} \). Given a message \( m \) and a public key \( pk \), \( \mathcal{E}_{pk}(m) \) produces a ciphertext \( c \) of \( m \). This algorithm may be probabilistic (involving random coins \( r \in \mathcal{R} \), and then denoted \( \mathcal{E}_{pk}(m; r) \)).
- **The decryption algorithm** \( \mathcal{D} \). Given a ciphertext \( c \) and the secret key \( sk \), \( \mathcal{D}_{sk}(c) \) 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 \( \text{Adv}^\text{ind}_{\pi}(A) \), as defined below where the adversary \( A \) is seen as a 2-stage Turing machine \( (A_1, A_2) \), should be negligible.

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

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 IND-CCA, also known as IND-CCA2, to be compared to IND-CCA1, which captures lunchtime attacks [17] only.
2.3 Secure Designs

The expected security level is thus IND-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 $\varphi_{pk}$ (and its inverse $\psi_{sk}$, 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 $\varphi_{pk}$, while the private key $sk$ defines the inverse $\psi_{sk}$ of $\varphi_{pk}$. Then,

$E_{pk}(m; r) = \varphi_{pk}(P(m, r)) \quad D_{sk}(c) = m, \text{ where } (m, r) = \mathcal{P}^{-1}(\psi_{sk}(c))$.

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 $\varphi$, 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 $A$ be any chosen-ciphertext adversary against $\varphi\text{-FDP}$, within time $\tau$. After $q_p$ and $q_d$ queries to the permutation oracles and the decryption oracle respectively,

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

where $T_\varphi$ is the time complexity for evaluating $\varphi$.

Let us briefly recall that for any algorithm $A$,

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

### 3.3 Sketch of the Proof

The goal of the proof is to simulate the oracles $P$, $P^{-1}$, and $D_{sk}$ 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 $P$ and $P^{-1}$ are first simulated by using a perfectly random permutation $P$ and its inverse $P^{-1}$. The last game provides a simulation of $D_{sk}$, without inverting $\varphi_{pk}$.

Anyway, the simulation remains almost perfect unless the adversary asks the pre-image via $\varphi_{pk}$ of the challenge ciphertext to the random permutation $P^{-1}$: it thus helps to invert $\varphi$. 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 $G$ and $H$ before encrypting with a trapdoor one-way permutation $\varphi_{pk}$. More precisely, for encrypting a message $m$, one randomly chooses $r$, and computes $s$ and $t$:

$$s = m \oplus G(r) \quad t = r \oplus H(s).$$

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

$$(s, t) = \psi_{sk}(c) \quad r = t \oplus H(s) \quad m = s \oplus G(r).$$
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 $\varphi_{pk}$). For that, we must simulate all the resources that the attacker can access, namely, the oracles $G$, $H$ but also the decryption oracle $D_{sk}$.

For the above 2-round OAEP, the decryption oracle does not seem simulatable. The following attack 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 $H$ to get respectively $h = H(s)$ and $h' = H(s')$. Then it chooses $t$ and computes $c = \varphi_{pk}(s, t)$. If it asks $c$ to $D_{sk}$, it gets the corresponding plaintext $m$. Then, it computes $t' = t \oplus h \oplus h'$ and $c' = \varphi_{pk}(s', t')$. If it asks $c'$ to $D_{sk}$, 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 $\varphi_{pk}$ from a set $E$ to a set $F$, while the private key is the inverse $\psi_{sk}$. 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 $\kappa$ such that:

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

However, note that in the case that $E \neq 0^\kappa \times \{0, 1\}^{k+\ell}$ 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 losing a factor 2 in efficiency, one can get $\kappa = 0$, with the iterated-RSA [2]).

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

$$F : \{0, 1\}^k \rightarrow \{0, 1\}^\ell \quad G : \{0, 1\}^\ell \rightarrow \{0, 1\}^k \quad H : \{0, 1\}^{k+\kappa} \rightarrow \{0, 1\}^\ell.$$

**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

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

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

$$s = u \oplus H(B\|t) \quad r = t \oplus G(s) \quad m = s \oplus F(r).$$

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 $\varphi$ [15].
Lemma 5. 

Proof (Full proof in the Appendix B.1). The main idea in simulating this game is that we make the components of the challenge \( c^* \) (namely \( r^*, f^*, s^*, g^*, t^*, h^*, u^* \) and \( c^* \)) independent to \( m^* \). We can do this by choosing ahead of time random values for \( r^*, s^*, \) and \( t^* \), 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 \( m^* \), the attacker has only the chance of one half to guess the bit \( b \). \( \Box \)

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.

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

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. \( \Box \)

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 \( c = \varphi_{pk}(B,t,h \oplus s) \). 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.

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

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 \( B \parallel t \), \( s \) or \( r \) are asked. For solving this problem, we modify the rule \text{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 \( \mathcal{H}(B \parallel t) \) is queried. \( \Box \)
**Game G₅:** We now complete the simulation of the oracle $D_{sk}$. We don’t ask any query to $\psi_{sk}$. 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 $G₄$.

- **Rule Decrypt-Init⁵**
  Look up for $(B, t, h) \in H$-List and $(s, g) \in G$-List such that $\varphi_{pk}(B, t, s \oplus h) = c$.
  - if the record is found, we furthermore define $u = s \oplus h$. 
  - otherwise, we take $B = \perp, t = \perp, u = \perp$.

- **Rule Decrypt-TS⁵**
  $r = t \oplus g, \quad f = F(r), \quad m = s \oplus f$.

The two games $G₅$ and $G₄$ 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 $G₄$:

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

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

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

where $\tau'$ 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 $GH$-List which contains all the tuples $(B, t, h, s, g, y)$ where $(B, t, h) \in H$-List, $(s, g) \in G$-List and $y = \varphi_{pk}(B, t, s \oplus h)$. 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 $A$ be any chosen-ciphertext adversary against the 3-round OAEP construction with the trapdoor permutation family $\varphi$, within time $\tau$. After $q_o$ and $q_d$ queries to the random oracles and the decryption oracle respectively,

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

where $T_{\varphi}$ is the time complexity for evaluating $\varphi$, 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.
A Security Proof of the $\varphi$-FDP Encryption

**Game 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 $\mathcal{D}_k$. 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 $c^*$ of $m^* = m_b$. This ciphertext comes from a random $\mathcal{R} \leftarrow \{0, 1\}^k$ and

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

In the second step, on input $c^*$, $A_2$ outputs a bit $b'$. We denote by $S_0$ the event $b' = b$ and use the same notation $S_0$ in any game $G_n$. Note that the adversary is given access to the decryption oracle $\mathcal{D}_k$ during both steps of the attack.
By definition,
\[ \varepsilon = \text{Adv}_n^{\text{ind-cca}}(A) = 2\Pr[S_0] - 1. \] (7)

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

**Game G_1:** A perfect simulation of the real attack game is described on the Figure 1. Actually, the rule Chal(1) simulates the way the challenge \(c^*\) is generated, exactly as the challenger would do. Besides, we simulate the two random permutation oracles \(P\) and \(P^{-1}\), and the decryption oracle \(D_{sk}\), by maintaining a list \(P\text{-List}\), and using a truly random permutation \(P\) and its inverse \(P^{-1}\).

Note that when we “store” an element \((m, r, p, y)\) in \(P\text{-List}\): if \(p \neq \bot\), and if there is already an element \((m, r, \star, y)\) in \(P\text{-List}\), we replace the latter by the former; if \(p \neq \bot\), 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 \(P\text{-List}\) is not used. The latter list is only introduced for later simulation in the proof.

\[ \Pr[S_1] = \Pr[S_0]. \] (8)

**Game G_2:** In this game, we modify the simulation of the oracle \(P\), 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 \(P\text{-List}\):
  - if the record is found,
    - if \(\alpha \neq \bot\), \(p = \alpha\);
    - otherwise, \(p \leftarrow \{0,1\}^{\ell+k}\).
  - otherwise, \(p = P(m, r)\).

Note that if \(\alpha = \bot\), we likely give a wrong answer (the correct answer should be \(\psi_{sk}(y)\)), we define this event \(BadP_2\). Unless this event happens, the two games \(G_2\) and \(G_1\) are perfectly indistinguishable:

\[ |\Pr[S_2] - \Pr[S_1]| \leq \Pr[BadP_2], \] (9)

**Game G_3:** We now modify the same way the simulation of the oracle \(P^{-1}\) by using the following rule:

- **Rule InvP(3)**
  
  Compute \(c = \varphi_{sk}(p)\) and look up for \((m, r, \alpha, c)\) in \(P\text{-List}\):
  - if the record is found, \((m, r)\) is defined,
  - otherwise we compute \((m, r) = P^{-1}(p)\).

The two games \(G_3\) and \(G_2\) are perfectly indistinguishable:

\[ \Pr[S_3] = \Pr[S_2] \quad \Pr[BadP_3] = \Pr[BadP_2]. \] (10)

**Game G_4:** In this game, we continue to modify the simulation of the oracles \(P\) and \(P^{-1}\), 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 \(P(m, r)\) by a random \(p\), and for a new \(p\) we answer \(P^{-1}(p)\) by a random pair \((m, r)\). We rewrite the rules as follows:

- **Rule EvalP(4)**
  
  Look up for \((m, r, \alpha, y)\) in \(P\text{-List}\):
  - if the record is found,
    - if \(\alpha \neq \bot\), \(p = \alpha\);
    - otherwise, \(p \leftarrow \{0,1\}^{\ell+k}\).
  - otherwise, \(p \leftarrow \{0,1\}^{\ell+k}\).
We have:

The two games $G_3$ and $G_4$ are perfectly indistinguishable unless a collision appears over $(m, r)$ or $p$ in $\mathcal{P}$-List, we define this event $\text{CollP}_4$. So, we have:

$$|\Pr[S_4] - \Pr[S_3]| \leq \Pr[\text{CollP}_4] \quad |\Pr[\text{BadP}_4] - \Pr[\text{BadP}_3]| \leq \Pr[\text{CollP}_4]. \quad (11)$$

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

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

**Game $G_5$:** Finally, we can simulate the decryption oracle $D_{sk}$ without inverting $\varphi_{sk}$:

**Rule InvP$^{(4)}$:**

Compute $c = \varphi_{sk}(p)$ and look up for $(m, r, \alpha, c)$ 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 $G_5$ and $G_4$ are perfectly indistinguishable since in case 2, we do exactly as $\mathcal{P}^{-1}$ was simulated by the rule InvP$^{(4)}$ in the previous game:

$$\Pr[S_5] = \Pr[S_4] \quad \Pr[\text{BadP}_5] = \Pr[\text{BadP}_4]. \quad (13)$$

**Game $G_6$:** 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 $\text{CollM}_6$.

$$|\Pr[S_6] - \Pr[S_5]| \leq \Pr[\text{CollM}_6] \quad |\Pr[\text{BadP}_6] - \Pr[\text{BadP}_5]| \leq \Pr[\text{CollM}_6]. \quad (14)$$

We see that for any $(m, r, \perp, y)$ except if $m = m^*$, $m$ 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,

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

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

$$\Pr[\text{BadP}_6] \leq \frac{q_p}{2^k}. \quad (16)$$

**Game $G_7$:** Now we suppress the element $(m^*, r^*, \perp, c^*)$ from $\mathcal{P}$-List.

**Rule ChalAdd$^{(7)}$:**

Do nothing.

The two games $G_7$ and $G_6$ are perfectly indistinguishable unless $(m^*, r^*)$ is asked for $\mathcal{P}$ (which event is included in event BadP$_7$, already excluded) or $p^*$ is asked to $\mathcal{P}^{-1}$. We define the latter event $\text{AskInvP}_7$. We have:

$$|\Pr[S_7] - \Pr[S_6]| \leq \Pr[\text{AskInvP}_7]. \quad (17)$$

Since $(m^*, r^*, p^*, c^*)$ 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 $b$:

$$\Pr[S_7] = \frac{1}{2}. \quad (18)$$

**Game $G_8$:** Instead of choosing $c^* = \varphi_{sk}(p^*)$, we choose $c^*$, uniformly at random.
Conclusion. Finally, combining all the above equations:

\[
\Pr[\text{AskInvP}_8] = \Pr[\text{AskInvP}_7].
\]  

(19)

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

\[
\Pr[\text{AskInvP}_8] \leq \text{Succ}_{w}^{ow}(\tau' + q_p \times T_\varphi),
\]

(20)

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

\[\varepsilon = \frac{1 + \epsilon}{2} - \frac{1}{2} = \Pr[\mathcal{S}_0] - \Pr[\mathcal{S}_7] \leq \Pr[\mathcal{S}_0] - \Pr[\mathcal{S}_1] + \Pr[\text{AskInvP}_7] + \Pr[\text{CollM}_6] + \Pr[\text{CollP}_4] + \Pr[\text{BadP}_2] \leq \Pr[\text{AskInvP}_8] + 2\Pr[\text{CollM}_6] + 2\Pr[\text{CollP}_4] + \Pr[\text{BadP}_6].\]

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

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

B Complements for the Proof of the Theorem 2

B.1 Proof of Lemma 3

Game \( G_{1,1} \): For proving this lemma, we present a more detailed sequence of games from the game \( G_1 \) to the game \( G_{1,2} \). We first make the value of the random seed \( r^* \) explicit and move its generation up-front.

\[\text{Rule Chal}^{(1,1)}\]

The two values \( r^+ \xleftarrow{\mathcal{R}} \{0,1\}^k \), \( f^+ \xleftarrow{\mathcal{R}} \{0,1\}^\ell \) have been chosen ahead of time, then

\[
\begin{align*}
    r^* &= r^+, & f^* &= f^+, & s^* &= m^+ \oplus f^+, & g^* &= \mathcal{G}(s^*), \\
    t^* &= r^+ \oplus g^*, & h^* &= \mathcal{H}(0^k||t^*), & u^* &= s^* \oplus h^*.
\end{align*}
\]

Compute \( c^* = \varphi_{pk}(0^k, t^*, u^*) \).

The two games \( G_{1,1} \) and \( G_1 \) are perfectly indistinguishable unless \( r^* \) has been asked for \( \mathcal{F} \). We define this event \( \text{AskF}_{1,1} \). We have:

\[
|\Pr[\mathcal{S}_{1,1}] - \Pr[\mathcal{S}_1]| \leq \Pr[\text{AskF}_{1,1}].
\]  

(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:

\[
\Pr[\mathcal{S}_{1,1}] = \frac{1}{2}.
\]  

(22)

Game \( G_{1,2} \): In this game, instead of defining \( s^* \) from \( f^+ \) which is a random value \( f^+ \), we randomly choose \( s^* \) and then we define \( f^+ \) from \( s^* \). Because \( s^* \) is chosen randomly, we give a random answer for the question \( s^* \) to \( \mathcal{G} \).
In this game, the two games $G_{1,2}$ and $G_{1,1}$ are perfectly indistinguishable unless $s^*$ is asked for $G$. We define this event $AskG_{1,2}$. We have:

$$|\Pr[AskF_{1,2}] - \Pr[AskF_{1,1}]| \leq \Pr[AskG_{1,2}].$$

(23)

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

$$\Pr[AskF_{1,2}] = \frac{q_f}{2e}.$$  

(24)

**Game $G_{1,3}$:** Similarly to the above game, instead of defining $t^*$ from a random $g^+$, we randomly choose $t^*$ and then we define $g^+$ from $t^*$. Because $t^*$ is chosen randomly, we give a random answer for the question $(0^e||t^*)$ to $H$.

**Rule 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

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

$$t^* = r^+ \oplus g^+ \quad h^* = H(0^e||t^*) \quad u^* = s^+ \oplus h^*.$$  

Compute $c^* = \varphi_{pk}(0^e, t^*, u^*)$.

The two games $G_{1,3}$ and $G_{1,2}$ are perfectly indistinguishable unless $0^e||t^*$ is asked for $H$. We define this event $AskH_{1,3}$. We have:

$$|\Pr[AskF_{1,3}] - \Pr[AskF_{1,2}]| \leq \Pr[AskH_{1,3}].$$

(25)

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

$$\Pr[AskF_{1,3}] = \frac{q_f}{2e}.$$  

**Game $G_{1,4}$:** We manufacture the challenge $c^*$ independently of anything else.

**Rule 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 $c^* = \varphi_{pk}(0^e, t^+, u^+)$.  

The distribution of $c^*$ remains the same:

$$\Pr[AskF_{1,4}] = \Pr[AskF_{1,3}].$$

(26)

**Game $G_{1,5}$:** We choose the challenge $c^*$ uniformly in the space $F$.

**Rule ChalC$^{(1,5)}$**

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

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

$$
\Pr[\text{AskH}_{1.5}] = \Pr[\text{AskH}_{1.5} \land \text{GoodB}] + \Pr[\text{AskH}_{1.5} \land \neg \text{GoodB}]
\geq \Pr[\text{AskH}_{1.5}|\text{GoodB}] \cdot \Pr[\text{GoodB}] \geq \Pr[\text{AskH}_{1.4}] \cdot \frac{1}{2^k}.
$$

(27)

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

$$
\Pr[S_1] \leq \Pr[S_{1.1}] + \Pr[\text{AskF}_{1.1}] \leq \frac{1}{2} + \Pr[\text{AskF}_{1.1}]
\leq \frac{1}{2} + \Pr[\text{AskF}_{1.2}] + \Pr[\text{AskG}_{1.2}]
\leq \frac{1}{2} + \Pr[\text{AskF}_{1.2}] + \Pr[\text{AskG}_{1.3}] + 2^k \cdot \Pr[\text{AskH}_{1.5}]
\leq \frac{1}{2} + \frac{q_f}{2^k} + \frac{q_d}{2^k} + 2^k \cdot \Pr[\text{AskH}_2].
$$

B.2 Proof of Lemma 4

**Game 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 $D$-query on $c = \varphi_{pk}(B, t, u)$ such that $\mathcal{H}(B\|t)$ has never been queried.

**Rule Decrypt-noT^{(2.1)}**

- Choose $h \sim R \{0, 1\}^t$ and set $s = u \oplus h$.
- Choose $g \sim R \{0, 1\}^k$ 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 $\text{G}_{2.1}$ and $\text{G}_2$ 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 $(q_f + q_d)/2^k$:

$$
|\Pr[\text{AskH}_{2.1}] - \Pr[\text{AskH}_2]| \leq q_d(q_f + q_d)/2^k.
$$

(28)

**Game 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 $h \sim R \{0, 1\}^t$ and set $s = u \oplus h$.
- Choose $g \sim R \{0, 1\}^k$ and set $r = t \oplus g$.
- Choose $f \sim R \{0, 1\}^f$ 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 $\text{G}_{2.2}$ and $\text{G}_{2.1}$ 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 $(q_f + q_d)/2^k$:

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

(29)

**Game 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 $m \xleftarrow{\$} \{0, 1\}^\ell$.
Choose $h \xleftarrow{\$} \{0, 1\}^k$ and set $s = u \oplus h$.
Choose $g \xleftarrow{\$} \{0, 1\}^k$ 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 $G_{2.3}$ and $G_{2.2}$ are perfectly indistinguishable:

$$
\Pr[\text{AskH}_{2.3}] = \Pr[\text{AskH}_{2.2}].
$$

Game $G_{2.4}$: 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 $c = \varphi_{pk}(B, t, u)$ 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 $g \xleftarrow{\$} \{0, 1\}^k$ and set $r = t \oplus g$.
Choose $f \xleftarrow{\$} \{0, 1\}^\ell$ and set $m = s \oplus f$.
Add $(r, f)$ in $\mathcal{F}$-List, $(s, g)$ in $\mathcal{G}$-List.

The two games $G_{2.4}$ and $G_{2.3}$ 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 $(q_f + q_d)/2^k$:

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

Game $G_{2.5}$: 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 $m \xleftarrow{\$} \{0, 1\}^\ell$.
Choose $g \xleftarrow{\$} \{0, 1\}^k$ 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 $G_{2.5}$ and $G_{2.4}$ are perfectly indistinguishable:

$$
\Pr[\text{AskH}_{2.5}] = \Pr[\text{AskH}_{2.4}].
$$

B.3 Proof of Lemma 5

Game $G_{3.1}$: 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-TnoT(3.1)

Choose $m \xleftarrow{\$} \{0, 1\}^\ell$.

Rule Decrypt-noT(3.1)

Choose $h \xleftarrow{\$} \{0, 1\}^\ell$.
Choose $m \xleftarrow{\$} \{0, 1\}^\ell$.
Add $(B, t, h)$ in $\mathcal{H}$-List.
\textbf{Rule EvalGAdd}\(^{(3.1)}\)

Search \((B, t, h) \in \mathcal{H}\)\-List and \((m, c) \in \mathcal{D}\)\-List such that \(c = \varphi_{pk}(B, t, h \oplus s)\). 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 \(G_{3.1}\) and \(G_{3}\) are perfectly indistinguishable unless \(r\) is asked to \(\mathcal{F}\) before \(s\) is asked to \(\mathcal{G}\), we denote this event by \(\text{AskRbS}\). 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 \(G_{3}\).

\[ \begin{align*}
| \Pr[\text{AskH}_{3.1}] - \Pr[\text{AskH}_3] | & \leq \Pr[\text{AskRbS}_{3.1}]. \\
\end{align*} \] (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 \(q_f/2^k\):

\[ \Pr[\text{AskRbS}_{3.1}] \leq q_d \cdot q_f/2^k. \] (34)

\textbf{Game \(G_{3.2}\):} We continue to simulate the oracle \(\mathcal{D}_{sk}\). We use the following rule:

\textbf{Rule Decrypt-noT}\(^{(3.2)}\)

Choose \(m \overset{R}{\longleftarrow} \{0, 1\}^\ell\).

In this game, we don’t store anymore \((B, t, h)\) in \(\mathcal{H}\)\-List. In the \(G_{3.1}\), for the question \(t\), we answer randomly \(h\), so the attacker in the two games \(G_{3.2}\) and \(G_{3.1}\) 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 \(G_{3.2}\) and \(G_{3.1}\) are perfectly indistinguishable unless \(s\) is asked to \(\mathcal{G}\) before \(B\|t\) is asked to \(\mathcal{H}\), we denote this event by \(\text{AskSbT}\). 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 \(G_{3.1}\).

\[ \begin{align*}
| \Pr[\text{AskH}_{3.2}] - \Pr[\text{AskH}_{3.1}] | & \leq \Pr[\text{AskSbT}_{3.2}]. \\
\end{align*} \] (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 \(q_g/2^k\):

\[ \Pr[\text{AskSbT}_{3.2}] \leq q_d \cdot q_g/2^\ell. \]
Figure 1. Formal Simulations of the IND-CCA Game against $\phi$-FDP
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 $\psi(s)$: if a record $(s, g)$ appears in $\mathcal{G}$-List, the answer is $g$. Otherwise the answer $g$ is chosen randomly: $g \in \{0, 1\}^k$ and the record $(s, g)$ is added in $\mathcal{G}$-List.

- **Rule EvalGAdd**

Query $\mathcal{H}(B, t, h)$: 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.

Query $\mathcal{D}(m, c)$: 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**

  Compute $(B, t, u) = \psi_m(c)$;

  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**

        $h = \mathcal{H}(B, ||t||)$,
        
        $s = u \oplus h$,
        
        $g = \mathcal{G}(s)$,
        
        $r = t \oplus g$,
        
        $f = \mathcal{F}(r)$,
        
        $m = s \oplus f$.
      
    - otherwise
      
      - **Rule Decrypt-TnoS**

      - same as rule Decrypt-TS.

  - otherwise
    
    - **Rule Decrypt-noT**

      same as rule Decrypt-TS.

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

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

- **Rule Chal**

  $f^* = \mathcal{F}(r^*)$,  
  $s^* = m^* \oplus f^*$,
  
  $g^* = \mathcal{G}(s^*)$,  
  $t^* = r^* \oplus g^*$,
  
  $h^* = \mathcal{H}(0^* || t^*)$,  
  $u^* = s^* \oplus h^*$.

  Compute $c^* = \varphi_{\text{pk}}(0^*, t^*, u^*)$.

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