Fujisaki-Okamoto IND-CCA hybrid encryption revisited

David Galindo, Sebastià Martín, Paz Morillo and Jorge L. Villar

Dep. Matemàtica Aplicada IV. Universitat Politècnica de Catalunya
Campus Nord, c/Jordi Girona, 1-3, 08034 Barcelona
e-mail: {dgalindo,sebasm,paz,jvillar}@mat.upc.es

Abstract

At Crypto'99, Fujisaki and Okamoto [10] presented a nice generic transformation from weak asymmetric and symmetric schemes into an IND-CCA hybrid encryption scheme in the Random Oracle Model. From this transformation, two specific candidates to standardization were designed: EPOC-2 [9] and PSEC-2 [16], based on Okamoto-Uchiyama and El Gamal primitives, respectively. Since then, several cryptanalysis of EPOC have been published, one in the Chosen Ciphertext Attack game and others making use of a poor implementation that is vulnerable to reject timing attacks.

The aim of this work is to avoid these attacks from the generic transformation, identifying the properties that an asymmetric scheme must hold to obtain a secure hybrid scheme. To achieve this, some ambiguities in the proof of the generic transformation [10] are described, which can lead to false claims. As a result the original conversion is modified and the range of asymmetric primitives that can be used is shortened.

In second place, the concept of *Easy Verifiable Primitive* is formalized, showing its connection with the Gap problems. Making use of these ideas, a *new* security proof for the modified transformation is given. The good news is that the reduction is *tight*, improving the concrete security claimed in the original work for the Easy Verifiable Primitives. For the rest of primitives the concrete security is improved at the cost of stronger assumptions.

Finally, the resistance of the new conversion against reject timing attacks is addressed.

Keywords: public-key cryptography, chosen-ciphertext security, tight reduction, Random Oracle Model, Okamoto-Uchiyama scheme, reject timing attacks.

1 Introduction

When developing a new public key encryption scheme there are two basic criteria that a designer wants to reach: *security* and *efficiency*. Security is obviously the main concern, and it is expressed in terms of an attacker's goal against the scheme and the means it uses. The standard security notion for a general purpose cryptosystem is *indistinguishability against adaptive chosen ciphertext attacks*, IND-CCA for short. Although there are other security notions equivalent to the latter (cf. [1, 19]), it is preferred for technical reasons. Proofs of security are accepted only if they are in the *provable security* model, in which security is polynomially reduced to trusted mathematical assumptions. Regarding efficiency, there are two main aspects to consider. On one hand, the computational complexity of the algorithms involved in the scheme and, on the other hand, the concrete security of the scheme, that is, how the security of the scheme is related to the computational assumptions which it is based on. There are other features of relevance, as the design simplicity or the length of the messages that can be encrypted.

However, to develop a practical provably secure cryptosystem in the sense of IND-CCA is a quite difficult task. In fact, few such schemes are known in the standard model, being the exceptions the schemes designed in the Cramer-Shoup paradigm [7]. In the idealized Random Oracle Model [2], several powerful generic constructions have been designed [10, 17, 14, 5], which provide practical IND-CCA schemes from weak asymmetric and symmetric schemes.

Among these constructions, [14, 5] present a better security reduction than [10, 17]. This is mainly due to the use of the *Plaintext Checking Oracle* introduced in [13]. The cost of using this oracle is that the security of the encryption scheme is in general based on (stronger) gap assumptions, when the asymmetric primitive is probabilistic.

In this paper we revisit the generic conversion by Fujisaki and Okamoto (FO) presented at Crypto'99. The particular instantiation of this conversion with the Okamoto-Uchiyama scheme [15], known as EPOC-2 [9], has found practical attacks that lead to a total break [12, 8, 18]. The most serious flaw was found in [12], where the secret key was recovered in the IND-CCA game itself. The authors of [12] pointed out that such a surprising result was related to the vagueness of the IND-CCA model when dealing with invalid ciphertexts. In the case of the original especification of EPOC-2, an attacker could obtain vital information about the system from that ciphertexts. The other attacks mentioned above ([8, 18]), take profit of extra information available at the real world, as the running time of the decryption algorithm. This enables to distinguish among the reasons to reject certain ciphertexts and it is used to launch an attack recovering again the secret key.

Our results. We incorporate the comments made by EPOC's authors in [12] about FO conversion. Then we show that some ambiguities still remain in the proof of security, with the outcome that the security result claimed in [10] cannot be guaranteed in general. This forces to slightly modify the conversion and to shorten the range of asymmetric primitives that can be used.

In second place, the concept of *Easy Verifiable Primitive* is formalized, and it is used to give a *new* security proof for the modified transformation. We show that the reduction is *tight*, improving the concrete security claimed in the original work for the Easy Verifiable Primitives. For the rest of primitives the concrete security is improved at the cost of a stronger assumption, that is, a gap assumption (see [13]).

Finally, the resistance of the new conversion against reject timing attacks is addressed. Since the vulnerability of a scheme against these attacks is closely related to the design of the rejection rules in the decryption algorithm, we take care about this when drawing the modification.

2 Preliminaries

In this section we recall some technical details and notations that are used in the rest of the paper.

Algorithmic notation. Assigning a value a to a variable x will be in general denoted by $x \leftarrow a$. Nevertheless, this notation can be extended to allow different meanings. If A is a non-empty set, then $x \leftarrow A$ denotes that x is uniformly chosen in A. If D is a probability distribution over A, then $x \leftarrow D$ means that x is chosen in A by sampling the distribution D. Finally, if $\mathcal{A}$ is an (probabilistic) algorithm, $x \leftarrow \mathcal{A}$ means that $\mathcal{A}$ is executed on some specified input and its (random) output is assigned to the variable x.

Negligible functions. The class of negligible functions on a parameter $\ell \in \mathbb{Z}^+$, denoted as $\mathsf{negl}(\ell)$, is the set of the functions $\epsilon : \mathbb{Z}^+ \rightarrow \mathbb{R}^+$ such that for any polynomial $p \in \mathbb{R}[\ell]$, there exist $C \in \mathbb{R}^+$ such that $\epsilon(\ell) < \frac{C}{p(\ell)}$, for all $\ell \in \mathbb{Z}^+$. Let $\mathsf{poly}(\ell)$ the class of functions $p : \mathbb{Z}^+ \rightarrow \mathbb{R}^+$ upper bounded in $\mathbb{Z}^+$ by some polynomial in $\mathbb{R}[\ell]$.

Set sequences. As usual, $\{0,1\}^\star$ and $\{0,1\}^\ell$ will respectively denote the set of all finite binary strings and the set of binary strings with length ℓ. A string set sequence, $X = \{X_\ell\}_{\ell\in\mathbf{Z}^+}$, is a *polynomial size* set if there exist a integer valued function $p_X(\ell) \in \mathsf{poly}(\ell)$ such that $X_\ell \subseteq \{0,1\}^{p_X(\ell)}$ for all $\ell \in \mathbb{Z}^+$. A polynomial size set, X, is *samplable* if there exists a probabilistic polynomial time algorithm (PPT) that on input 1^ℓ, outputs an uniformly distributed random element in X_ℓ. Moreover, X is *recognizable* if there exist a polynomial time algorithm (PT) that on input 1^ℓ and a string s, with size polynomial in ℓ, outputs 1 if and only if $s \in X_\ell$. The cardinality of a set sequence A (as a function of ℓ) will be denoted by $|A|$.

These notions can be easily extended to non-strings sets by using polynomial size injective encoding maps. In the sequel, the use of natural encodings (e.g. binary representations of integers) is assumed when necessary.

Hereafter, the word 'sequence' in 'set sequence', 'map sequence' and 'probability distribution sequence' will be omitted.

Keypair generators. Let PK and SK polynomial size sets such that the sets PK_ℓ are all disjoint and there exists a PT algorithm that on input $sk \in SK_\ell$ outputs an element $pk \in PK_\ell$. Suppose that there also exists a PT algorithm that on input $pk \in PK_\ell$ outputs ℓ. Let I a polynomial time samplable probability distribution over $PK \times SK$. The triple (PK, SK, I) will be called a keypair generator.

Set and map families. Given a keypair generator, the family $\{X_{pk}\}_{pk\in PK}$ is referred as the *set family* X. In the same way, a *map family* f is defined as $\{f_{pk} : X_{pk} \rightarrow Z_{pk}\}_{pk\in PK}$. Given a set family, X, the cardinality $|X|$ as a function of ℓ is defined to be the minimal value of $|X_{pk}|$, where $pk \in PK_\ell$.

A set family X is *recognizable* if there exist a PT algorithm that on input $pk \in PK$ and a string s, with size polynomial in ℓ, outputs 1 if and only if $s \in X_{pk}$. A conjectured counterexample is the set family $X_{pk} = Q_n$ of the quadratic residues modulo $n = pq$ (p, q different primes with length ℓ). However, if $sk = (p,q)$ is also provided then there exist an efficient way to recognise the elements in Q_n.

3 Easy verifiable functions

Firstly, recall the definition of a trapdoor one-way function family.

Definition 1 *Let (PK, SK, I) a keypair generator. Let X and Z be polynomial size set families. Let $f : X \to Z$ be a family of injective maps and $g = \{g_{sk} : Z_{pk} \to X_{pk}\}_{sk\in SK}$ the family of their inverses, i.e. $g_{sk}(f_{pk}(x)) = x$ for all possible pairs (pk, sk) generated by I and for all $x \in X_{pk}$. The map family, f, is called a* Trapdoor One-Way (TOW) *function family (with respect to the probability distribution I) if and only if*

1. *there exist a PT algorithm that on input (pk, x) outputs $f_{pk}(x)$ for all $pk \in PK$ and $x \in X_{pk}$.*
2. *there exist a PT algorithm that on input (sk, z) outputs $g_{sk}(z)$ for all $sk \in SK$ and $z \in Z_{pk}$.*
3. *for any PPT algorithm $\mathcal{A}_{\mathsf{OW}(f)}$,*

$$\mathsf{Pr}\left[\mathcal{A}_{\mathsf{OW}(f)}(pk, f_{pk}(x)) = x \mid (pk, sk) \leftarrow I_\ell;\ x \leftarrow X_{pk}\right] \in \mathsf{negl}(\ell)$$

The following definition, based on [17], is somewhat related to the notion of probabilistic one-way encryption.

Definition 2 *Let (PK, SK, I) a keypair generator. Let X, Y and Z be polynomial size set families. Let $f : X \times Y \to Z$ be a family of injective maps and $g = \{g_{sk} : Z_{pk} \to X_{pk}\}_{sk\in SK}$ the family of their partial inverses, i.e. $g_{sk}(f_{pk}(x, y)) = x$ for all possible pairs (pk, sk) generated by I and for all $x \in X_{pk}$ and $y \in Y_{pk}$. The map family, f, is called a* Trapdoor Partial One-Way (TPOW) *function family (with respect to the probability distribution I) if and only if*

1. *there exist a PT algorithm that on input (pk, x, y) outputs $f_{pk}(x, y)$ for all $pk \in PK$, $x \in X_{pk}$ and $y \in Y_{pk}$.*
2. *there exist a PT algorithm that on input (sk, z) outputs $g_{sk}(z)$ for all $sk \in SK$ and $z \in Z_{pk}$.*
3. *for any PPT algorithm $\mathcal{A}_{\mathsf{POW}(f)}$,*

$$\mathsf{Pr}\left[\mathcal{A}_{\mathsf{POW}(f)}(pk, f_{pk}(x, y)) = x \mid (pk, sk) \leftarrow I_\ell;\ x \leftarrow X_{pk};\ y \leftarrow Y_{pk}\right] \in \mathsf{negl}(\ell)$$

The last condition can be reformulated in terms of the game

```
Game POW()
  $(pk, sk) \leftarrow I_\ell$
  $x \leftarrow X_{pk};\ y \leftarrow Y_{pk}$
  $x' \leftarrow \mathcal{A}_{\mathsf{POW}(f)}(pk, f_{pk}(x, y))$
```

and the probability $\mathsf{Succ}\left[\mathcal{A}_{\mathsf{POW}(f)}\right] = \mathsf{Pr}\left[x' = x\right] \in \mathsf{negl}(\ell)$.

Notice that the concept of TOW function family can be seen as a particular case of TPOW, in which $|Y| = 1$.

Starting from f, a probabilistic one-way cryptosystem, $(\mathsf{KeyGen}^f, \mathsf{Enc}^f, \mathsf{Dec}^f)$, is obtained in the following way: the keys $(pk, sk) = \mathsf{KeyGen}^f(1^\ell) = I_\ell$ are generated by using the sampling algorithm for I, the ciphertext for a message $x \in X_{pk}$ with randomness $y \leftarrow Y_{pk}$ is $c = \mathsf{Enc}^f(pk, x) = f_{pk}(x, y)$ and a valid ciphertext $z \in Z_{pk}$ is decrypted by means of $\mathsf{Dec}^f(sk, c) = g_{sk}(c)$. Note that we are implicitly assuming that Y is samplable. In this context, there is no need to specify what happens when an invalid ciphertext, i.e. a polynomial size string $z \notin Z_{pk}$, is submited to Dec^f.

A new kind of attacks and computational problems have been introduced and found various applications in the context of probabilistic cryptosystems (cf [13, 14]). In this new scenario, the attacker has access to a *Plaintext-Checking Oracle* that checks if a given ciphertext z is an encryption of a given message x. This attack is called Plaintext-Checking Attack (PCA), and it can be reformulated in terms of trapdoor partial one-way functions.

Definition 3 *A TPOW function family $f : X \times Y \to Z$ is* Partial One-Way against Plaintext-Checking Attacks (POW-PCA) *if it is a TPOW function even when access to a plaintext checking oracle $\mathcal{C}_{pk}$ is given. For a query (x, z), where $x \in X_{pk}$ and $z \in Z_{pk}$, $\mathcal{C}_{pk}$ answers 1 if there exists $y \in Y_{pk}$ such that $f_{pk}(x, y) = z$, and 0 otherwise. (It is assumed that if x or z are outside their domains, the oracle also answers 0.)*

This notion is stronger than partial one-wayness, since now the adversary is provided with extra computational resources. Now we formalize the concept of *easy verifiability*, informally described in [17], that captures the situation in which there exists an efficient algorithm that *verifies* if a pair (x, z) is correct, that is, the algorithm implements a plaintext checking oracle.

Definition 4 *The TPOW map family $f : X \times Y \to Z$ is* easy verifiable *if and only if there exists a (deterministic) PT algorithm $\mathcal{V}$, called* plaintext checking algorithm*, such that given any $pk \in PK$, $\mathcal{V}(pk, x, z) = 1$ if $x \in X_{pk}$, $z \in Z_{pk}$ and there exists $y \in Y_{pk}$ such that $f_{pk}(x, y) = z$, and 0 otherwise.*

Obviously, if f is easy verifiable then the Plaintext-Checking Oracle for f can be replaced by the algorithm $\mathcal{V}$, without introducing any modification in the adversary's model of computation. These functions are very interesting, since

Lemma 5 *If the map family $f : X \times Y \to Z$ is easy verifiable then it is POW-PCA.*

Note that the existence of the plaintext checking algorithm would imply that the set X is recognizable. This is due to the fact that the polynomial time algorithm that computes $f_{pk}(x, y)$ could work when x lies in a set $\bar{X}_{pk}$ broader than X_{pk}. Then, by definition, the output of $\mathcal{V}(pk, x, f_{pk}(x, y))$, for any choice of $y \in Y_{pk}$, can be used to efficiently recognise if $x \in \bar{X}_{pk}$ is in X_{pk}. For instance, no easy verifiable function exists if $X_{pk} = Q_n$ (i.e. the set of quadratic residues modulo $n = pq$) unless the quadratic residuosity modulo n problem is solvable in deterministic polynomial time. Nevertheless, such a function can achieve POW-PCA under a computational gap assumption (e.g. the gap between the quadratic residuosity modulo n and the factoring n assumptions).

It is straightforward to modify a TOW function family $f' : X \to Z'$ to obtain a easy verifiable function family f. To do it, simply take $Y = \{0,1\}^{p(\ell)}$, where $p(\ell) \in \mathsf{poly}(\ell)$, and define $f_{pk}(x, y) = (f'_{pk}(x), y)$, that is, leaving y "in the clear".

For an arbitrary TPOW function a plaintext checking algorithm could not exist. For instance, this is supposed to be the case for El Gamal and Okamoto-Uchiyama functions. In this situation, we are forced to base POW-PCA on a gap problem, which is a stronger assumption (cf [13, 14]).

A non-trivial example of easy verifiable function is the RSA-Paillier trapdoor bijection defined in [4]. A generalization of that function is presented below.

3.1 Non-trivial families of easy verifiable functions

Let $n = pq$, where p and q are different primes with equal length ℓ. Let $e < n$ be an integer such that $\gcd(e, (p-1)(q-1)) = 1$. For any integer $r > 1$ with size polynomial in ℓ, consider the subset $\Omega_{n,r} \subset \mathbb{Z}_{nr}$ defined as $\Omega_{n,r} = \mathbb{Z}_n^\star + n\mathbb{Z}_r$. Then, the function

$$\begin{array}{rcl} f_{n,r,e} : \mathbb{Z}_n^\star \times \mathbb{Z}_r & \longrightarrow & \Omega_{n,r} \\ (x, y) & \longrightarrow & x^e + ny \bmod nr \end{array}$$

is a trapdoor bijection family, for $pk = (n, r, e)$ and $sk = (p, q, r, d)$, where d is the inverse of e modulo $(p-1)(q-1)$.

Notice that this function is well defined since $z \in \Omega_{n,r}$ iff $z \bmod n \in \mathbb{Z}_n^\star$. Let see that $f_{n,r,e}$ is a bijection. Suppose that $f_{n,r,e}(x_0, y_0) = f_{n,r,e}(x_1, y_1)$ for some x_0, y_0, x_1 and y_1. Reducing the equality modulo n we get $x_0^e = x_1^e \bmod n$, and then $x_0 = x_1 \bmod n$. This implies $ny_0 = ny_1 \bmod nr$, so $y_0 = y_1 \bmod r$ and the function $f_{n,r,e}$ is injective. Finally, given (p, q, r, d), to invert $f_{n,r,e}$ on input $z = f_{n,r,e}(x, y)$, it suffices to compute $x = z^d \bmod n$. Then, y is easily obtained from the equation $ny = z - x^e \bmod nr$. This shows $f_{n,r,e}$ is exhaustive, and therefore it is a bijection.

The above implies there exist two PT algorithms that compute both $f_{n,r,e}$ and its partial inverse.

Proposition 6 *The partial one-wayness of the bijection family $f_{n,r,e}$ is tightly equivalent to the one-wayness of $RSA[n, e]$.*

Proof:
$\Rightarrow$) Assume that for some ℓ there exist a PPT algorithm, $\mathcal{A}$, breaking the partial one-wayness of $f_{n,r,e}$ in time T and probability ϵ, i.e.

$$\mathsf{Pr}\left[\mathcal{A}(n, r, e, x^e + ny \bmod nr) = x \mid x \leftarrow \mathbb{Z}_n^\star; y \leftarrow \mathbb{Z}_r\right] = \epsilon$$

The following PPT algorithm, $\mathcal{B}$, can be used to invert the $RSA[n, e]$ function (i.e. $RSA[n, e](x) = x^e \bmod n$) in time $T + O(\ell^2)$ with probability at least ϵ:

```
$\mathcal{B}(n, e, z)$
1  $y \leftarrow \mathbb{Z}_r$, $z' = z + ny \bmod nr$
2  $x \leftarrow \mathcal{A}(n, r, e, z')$
3  return $x$
```

Then, $\mathsf{Pr}\left[\mathcal{B}(n, x^e \bmod n) = x \mid x \leftarrow \mathbb{Z}_n^\star\right] \geq \epsilon$.
$\Leftarrow$) Trivial. ■

Proposition 7 *The bijection family $f_{n,r,e}$ is easy verifiable.*

Proof: Given (n, r, e), it is straightforward to design a plaintext checking algorithm. Firstly, verify if $x \in \mathbb{Z}_n^\star$ and $z \in \Omega_{n,r}$, that is, $z < nr$ and $z \bmod n \in \mathbb{Z}_n^\star$. Then, check if the equation $x^e \equiv z \pmod{n}$ holds. ■

4 Encryption security

Let us briefly recall some standard definitions about the security of both symmetric and asymmetric encryption.

4.1 Symmetric encryption

Let K and M be two (samplable and recognisable) polynomial size sets that respectively denote the key and message spaces. Consider a symmetric encryption scheme $\mathcal{E}^{sym} = (\mathsf{KeyGen}^{sym}, \mathsf{Enc}^{sym}, \mathsf{Dec}^{sym})$ over these sets with the following properties:

- KeyGen^{sym} is a PPT algorithm that on input 1^ℓ outputs an uniformly distributed element in K_ℓ.
- Enc^{sym} and Dec^{sym} are PT algorithms with inputs are in $K_\ell \times M_\ell$ and outputs in M_ℓ. Denote $\mathsf{Enc}_k^{sym}(m) = \mathsf{Enc}^{sym}(k, m)$ and $\mathsf{Dec}_k^{sym}(c) = \mathsf{Dec}^{sym}(k, c)$. For each $k \in K_\ell$, Enc_k^{sym} is a bijection on M_ℓ and Dec_k^{sym} is its inverse.
- For each pair $(m, c) \in M_\ell \times M_\ell$ there are at most γ values of $k \in K_\ell$ such that $c = \mathsf{Enc}_k^{sym}(m)$. (Most of the known symmetric cryptosystems have $\gamma = 1$.)

Such a cryptosystem has *indistinguishability of encryptions* (IND-SYM), also called Find-Guess security in [10], if any couple of PPT algorithms $\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})} = (\mathcal{A}_1, \mathcal{A}_2)$ (called “finding” and “guessing” stages of the adversary) have negligible advantage in the following game:

```
Game IND-SYM()
1  $b \leftarrow \{0,1\}$
2  $(m_0, m_1, s) \leftarrow \mathcal{A}_1(1^\ell)$
3  $k \leftarrow K_\ell$; $c^\star = \mathsf{Enc}_k^{sym}(m_b)$
4  $b' \leftarrow \mathcal{A}_2(s, c^\star)$
```

That is, $\mathcal{E}^{sym}$ is IND-SYM if and only if for all $(\mathcal{A}_1, \mathcal{A}_2)$,

$$\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right] = \left|2\mathsf{Pr}\left[b' = b\right] - 1\right| = \left|\mathsf{Pr}\left[b' = b\right] - \mathsf{Pr}\left[b' \neq b\right]\right| \in \mathsf{negl}(\ell)$$

The messages m_0 and m_1 generated by $\mathcal{A}_1$ must be in M_ℓ.

4.2 Asymmetric encryption

Let (PK, SK, I) a keypair generator, defined as in section 2. Let M, R and C be polynomial size set families. Consider an asymmetric encryption scheme $\mathcal{E}^{sym} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ over these sets with the following properties:

- KeyGen is a PPT sampling algorithm for I, that is, $(pk, sk) = \mathsf{KeyGen}(1^\ell)$ is distributed as I_ℓ on $PK_\ell \times SK_\ell$.
- $\mathsf{Enc} : PK \times M \times R \to C$ and $\mathsf{Dec} : SK \times C \to M$ are PT algorithms such that for any pair (pk, sk) generated by $\mathsf{KeyGen}(1^\ell)$, $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m, r)) = m$ for any $m \in M_{pk}$ and $r \in R_{pk}$. In fact, Enc can be seen as a PPT algorithm with input in $PK \times M$ and coins taken in R.

Such a cryptosystem has *indistinguishability of encryptions under a chosen ciphertext attack* (IND-CCA), if any couple of PPT algorithms $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})} = (\mathcal{A}_1, \mathcal{A}_2)$ have negligible advantage in trying to distinguish the encryptions of two (conveniently selected) messages, with access to two decryption oracles $\mathcal{D}_{sk}$ and $\mathcal{D}_{sk,c^\star}$. When queried with a ciphertext c the first decription oracle answers $\mathsf{Dec}(sk, c)$. The only difference between $\mathcal{D}_{sk}$ and $\mathcal{D}_{sk,c^\star}$ is that $\mathcal{D}_{sk,c^\star}$ rejects the query $c^\star$, answering the special reject symbol $\perp$.

More formally, consider the following game:

```
Game IND-CCA()
1  $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell)$
2  $b \leftarrow \{0,1\}$
3  $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{\mathcal{D}_{sk}}(pk)$
4  $c^\star \leftarrow \mathsf{Enc}(pk, m_b)$
5  $b' \leftarrow \mathcal{A}_2^{G,H,\mathcal{D}_{sk,c^\star}}(s, c^\star)$
```

Then, $\mathcal{E}$ is IND-CCA if and only if for all couples of PPT algorithms $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}\right] = \left|2\mathsf{Pr}\left[b' = b\right] - 1\right| = \left|\mathsf{Pr}\left[b' = b\right] - \mathsf{Pr}\left[b' \neq b\right]\right| \in \mathsf{negl}(\ell)$$

The messages m_0 and m_1 generated by $\mathcal{A}_1$ must be in M_{pk}.

Notice that the decryption oracle formalizes the access to a decryption machine. Thus, the adversary is free to submit any polynomially bounded string (except for the target ciphertext, $c^\star$, in the guessing stage) to this oracle. This means that IND-CCA security depends not only on the encryption algorithm but also on the concrete implementation of the decryption algorithm, including its behaviour for inputs outside the set of valid ciphertexts (i.e. ciphertexts of the form $\mathsf{Enc}(pk, m, r)$ for $m \in M_{pk}$ and $r \in R_{pk}$). This behaviour can give very useful information for an adversary.

4.3 Random oracle model

There are some ways to define random functions or random oracles in the literature. In the seminal work [2], random oracles act as random functions from $\{0,1\}^\star$ to $\{0,1\}^\infty$, while in the influential paper [3] the random functions are collections of functions. Moreover, in the second definition for a given value of the complexity parameter, ℓ, the corresponding function goes from $\{0,1\}^{p_I(\ell)}$ to $\{0,1\}^{p_O(\ell)}$, where $p_I(\ell), p_O(\ell) \in \mathsf{poly}(\ell)$. Furthermore, the security of some schemes (e.g. [11]) depends on which definition of random function is used.

In this paper, a random oracle is viewed as a special type of random process or random sequence. The random oracle is defined through its idealised functionality, that is closely related to the random oracle simulations often used in the proofs of security.

Let A be a samplable polynomial size set. A *random function* G over A is a sequence of uniformly distributed independent random variables over A, indexed by the elements of $\{0,1\}^\star$. Notice that $\{0,1\}^\star$ is an ordered set. A *random oracle* over A is an oracle that answers queries exactly as if the random function G was evaluated.

The main property of a random function is that the joint distribution of q_G variables $G(s)$ for distinct values of s is the same regardless which values of s are selected. Thus, an efficient probabilistic (interactive) algorithm can simulate this random function by means of a table $\mathcal{T}_G$ storing all previous queries along with their answers. Any new (yet unanswered) query will be answered with a "fresh" random value, that will be annotated in $\mathcal{T}_G$.

```
$G(s)$
1  if $s \in \mathcal{T}_G$; return $\mathcal{T}_G(s)$; endif
2  $g \leftarrow A$
3  insert $(s, g)$ in table $\mathcal{T}_G$
4  return $g$
```

Here, $s \in \mathcal{T}_G$ will denote the fact that s has been queried to G by some party and $\mathcal{T}_G(s)$ will denote the answer given by G to that query.

Notice that the above algorithm runs in polynomial time and space if,

- A is a samplable polynomial size set
- during the game, in which the different parties have access to the random oracle, no more than a polynomial quantity, $q_G(\ell) \in \mathsf{poly}(\ell)$, of queries are made
- the size of each query is limited by a polynomial function in ℓ

This last condition will be obviously fulfil if all parties are modeled by polynomial time machines.

Finally, IND-CCA security in the Random Oracle Model (ROM) of an asymmetric cryptosystem $\mathcal{E}$ is defined in the same way as above, but providing the adversary with oracle access to one or more random functions. In order to formalize the random coins of the random functions, a step $G \leftarrow \mathcal{R}(A)$ will be added at the beginning of the IND-CCA game for each random function used.

Obviously, in the real world random functions have to be adequately replaced by (hash) function families with a polynomial size description. This description will be included in the public data available to all parties in a protocol (e.g. the public key of an encryption scheme).

5 Revisiting Fujisaki-Okamoto hybrid scheme

In this section, the transformation introduced in [10] of weak symmetric and asymmetric schemes into an IND-CCA hybrid encryption scheme is revisited.

5.1 The original construction

Let $\mathcal{E}^f = (\mathsf{KeyGen}^f, \mathsf{Enc}^f, \mathsf{Dec}^f)$ be a probabilistic asymmetric encryption scheme, defined from a TPOW function family f over the sets X, Y and Z, and $\mathcal{E}^{sym} = (\mathsf{KeyGen}^{sym}, \mathsf{Enc}^{sym}, \mathsf{Dec}^{sym})$ be a symmetric encryption scheme over the sets K and M. Let G be a random function over K and H an independent random function over Y. The hybrid scheme, $\mathcal{E}^{FO} = (\mathsf{KeyGen}^{FO}, \mathsf{Enc}^{FO}, \mathsf{Dec}^{FO})$, proposed by Fujisaki and Okamoto works as follows.

Key generation. The public and secret keys are generated as in KeyGen^f.

Encryption. The ciphertext for a message $m \in M_\ell$ is $c = (f_{pk}(x,y), \mathsf{Enc}^{sym}_{G(x)}(m))$, where $y = H(x,m)$ and x is uniformly chosen in X_{pk}.

Decryption. To decrypt a ciphertext $c = (c_1, c_2)$, firstly compute $x = g_{sk}(c_1)$. Then, compute $m = \mathsf{Dec}^{sym}_{G(x)}(c_2)$ and return m if $c_1 = f_{pk}(x, H(x,m))$. Otherwise, return the reject symbol $\perp$. If it is not possible to compute $g_{sk}(c_1)$ or $\mathsf{Dec}^{sym}_{G(x)}(c_2)$, return $\perp$.

Let $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}[T, \epsilon, q_G, q_H, q_D]$ denote an adversary against the IND-CCA security of the above cryptosystem that runs in time T with advantage ϵ, doing no more than q_G, q_H and q_D queries respectively to the random oracles G, H and to the decryption oracle, $\mathcal{D}_{sk}$.

Theorem 8 *If there exists for some values of ℓ an adversary $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}[T, \epsilon, q_G, q_H, q_D]$, then there exist an adversary $\mathcal{A}_{\mathsf{POW}(f)}$ against the POW of f in time T_1 with success probability ϵ_1 and an adversary $\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}$ against the IND-SYM security of $\mathcal{E}^{sym}$ in time T_2 with advantage ϵ_2 such that*

$$\epsilon \leq (2(q_G + q_H)\epsilon_1 + \epsilon_2 + 1)\left(1 - 2\epsilon_1 - 2\epsilon_2 - \frac{1}{|Y|} - \frac{1}{|M|}\right)^{-q_D} - 1$$

and

$$T = \min(T_1, T_2) - O((q_G + q_H)\log(|X||M|))$$

The main drawback of this scheme is that the security reduction obtained in the proof is not tight, due to the quantity $q_G + q_H$ multiplying ϵ_1. However, the same authors improved in [11] this result for the particular case of the Okamoto-Uchiyama scheme [15] (known as EPOC-2) and claimed, without proof, that a tight reduction is obtained for *trivial* easy verifiable primitives, using our terminology.

5.2 Identifying dangerous ambiguities

However, as pointed out in the introduction, several attacks against EPOC-2 have been found [12, 8, 18]. Despite the precisions introduced about FO conversion after [12], there are still some ambiguities in the scheme, as well as in the security proof, that compromise the validity of the theorem above.

For instance, let us consider a TPOW function family f, and $X_{pk} \subset \bar{X}_{pk}$ such that $f_{pk}(x,y)$ is computable in polynomial time for any $x \in \bar{X}_{pk}$ and $y \in Y_{pk}$. Then, some badly generated ciphertexts $c = (f_{pk}(x, H(x,m)), \mathsf{Enc}^{sym}_{G(x)}(m))$ for $x \in \bar{X}_{pk} \setminus X_{pk}$ may be accepted. This was the case for Okamoto-Uchiyama function in the original EPOC-2, where $\bar{X}_{pk} = \mathbb{Z}_{2^\ell+1}$ and $X_{pk} = \mathbb{Z}_{2^\ell}$, for $2^\ell < p < 2^\ell + 1$. This information was used in [12] to obtain the secret value p.

As Fujisaki and Okamoto proposed later in [11], this attack is avoided if all ciphertexts (c_1, c_2) such that $g_{sk}(c_1) \notin X_{pk}$ are rejected. However, when this change is included in the general conversion a problem of different kind arises. If X is not a recognizable set, the checking cannot be performed in polynomial time. In this case the simulation of the $\mathcal{D}_{sk}$ in the proof is not correct.

Yet one could manage to use an additional oracle to solve this problem. In this situation, an adversary can use the decryption oracle to solve a *difficult* decisional problem. As a result, we only could guarantee that breaking security of the cryptosystem is equivalent to solve a gap problem, that is, a stronger assumption than claimed.

This is the case for the Blum-Williams one-way trapdoor bijection family (i.e. squaring quadratic residues modulo $n = pq$), where $X_{pk} = Q_n$ and $\bar{X}_{pk} = \mathbb{Z}_n$. Rejection of all ciphertexts (c_1, c_2) such that $g_{sk}(c_1) \notin X_{pk}$ means that the adversary will know if an arbitary $x \in \mathbb{Z}_n$ is a quadratic residue. Thus, the $\mathsf{IND\text{-}CCA}$ security of the hybrid cryptosystem will be based on the gap between the quadratic residuosity modulo n and factoring n assumptions.

5.3 The new proposal

From the above discussion we know that, although it is necessary to check if $g_{sk}(c_1) \in X_{pk}$, to avoid leaking vital information, this cannot be done in all cases. In this section we restrict the asymmetric primitives to that which admit a correct and unambiguous proof of security for the general transformation.

We also take into account the results in [8, 18] that use the ability to distinguish among rejection rules in the hybrid scheme to launch a total break. Thus, we slightly modify the specification of the decryption algorithm in the conversion.

Finally, the recent developments in [14, 5, 6] can be applied to this transformation, and together with the concept of easy verifiable primitives, they are used to give a *new proof of security* improving the concrete security result presented in the original work.

As in the original transformation, let $\mathcal{E}^f = (\mathsf{KeyGen}^f, \mathsf{Enc}^f, \mathsf{Dec}^f)$ be a probabilistic asymmetric encryption scheme, defined from a TPOW function family f over the sets X, Y and Z, and $\mathcal{E}^{sym} = (\mathsf{KeyGen}^{sym}, \mathsf{Enc}^{sym}, \mathsf{Dec}^{sym})$ be a symmetric encryption scheme over the sets K and M. Let G be a random function over K and H an independent random function over Y.

The first change we introduce is that the random functions G and H are defined with unrestricted inputs, as explained in subsection 4.3. We think it is not realistic to restrict the inputs of the random functions, as suggested in [10], since in a practical implementation random functions are replaced by cryptographic hash functions. Then, if a proof of security can be driven for unrestricted domains, this choice is preferable.

The following modification to the original proposal shorten the range of asymmetric primitives that can be used. Now, X and M must be recognizable sets. This not an actual restriction for M, since almost always $M_\ell = \{0,1\}^{p(\ell)}$, for some polynomial p. It is not demmanded Z to be a recognizable set. Instead of this, it is assumed that there exists a recognizable set $\bar{Z}$ such that $Z_{pk} \subseteq \bar{Z}_{pk}$, and that the partial inverse of f_{pk} can also be computed (in polynomial time) on elements of the extended set $\bar{Z}_{pk}$.

The proposed hybrid cryptosystem, $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, is almost the same as the original. The only but important change is that now two different reject symbols are produced in the decryption algorithm Dec. Thus, when a ciphertext is rejected, the adversary will know the reason, obtaining different reject symbols without mounting a timing attack. Then, if the computing time of each step in the algorithm is independent of the data, the scheme is closely to be robust against reject timing attacks.

```
Dec($sk, c$)
1  if $c \notin \bar{Z}_{pk} \times M_\ell$; return $\perp_1$; endif
2  $(c_1, c_2) = c$
3  $x \leftarrow g_{sk}(c_1)$
4  $m \leftarrow \mathsf{Dec}^{sym}_{G(x)}(c_2)$
5  $y \leftarrow H(x, m)$
6  if $x \notin X_{pk}$ or $f_{pk}(x, y) \neq c_1$; return $\perp_2$; endif
7  return $m$
```

We point out that in the or operation in step 6 of the algorithm both predicates have *always* to be evaluated, in order to prevent the adversary to detect an extra rejection reason.

Now, the security results are stated. The first theorem is for the special case when f is an easy verifiable function family, while the second theorem works for general TPOW function families.

Theorem 9 *If there exists for some values of ℓ an adversary $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}[T, \epsilon, q_G, q_H, q_D]$ against the IND-CCA security of the proposed cryptosystem, then there exist an adversary $\mathcal{A}_{\mathsf{POW}(f)}$ that in time T_1 breaks the POW of f with success probability ϵ_1 and an adversary $\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}$ that in time T breaks IND-SYM security of $\mathcal{E}^{sym}$ with advantage ϵ_2 such that*

$$\epsilon \leq \epsilon_1 + 3\epsilon_2 + \frac{2q_D q_H \gamma}{|K| - q_D q_H \gamma} + \frac{2q_D}{|Y| - q_D}$$

and

$$T_1 \leq (q_G + q_H + q_D + q_G q_D) T[\mathcal{V}] + q_D \Big(T[f] + T[\mathsf{Dec}^{sym}] \Big) + T$$

where $T[\mathcal{V}]$ is the time complexity of the plaintext checking algoritm for f and $T[f]$ is the time complexity of f.

Proof: The proof is delayed to the appendix. ■

Notice that now the probabilities are tightly related.

In the general case, there could not exist the plaintext checking algorithm. Using the access to a plaintext checking oracle instead, the following result is straightforward.

Corollary 10 *If there exists for some values of ℓ an adversary $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}[T, \epsilon, q_G, q_H, q_D]$ against the IND-CCA security of the proposed cryptosystem, then there exist an adversary $\mathcal{A}_{\mathsf{POW-PCA}(f)}$ that in time T_1 breaks the POW-PCA of f with success probability ϵ_1 and an adversary $\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}$ that in time T breaks IND-SYM security of $\mathcal{E}^{sym}$ with advantage ϵ_2 such that*

$$\epsilon \leq \epsilon_1 + 3\epsilon_2 + \frac{2q_D q_H \gamma}{|K| - q_D q_H \gamma} + \frac{2q_D}{|Y| - q_D}$$

and

$$T_1 \leq (q_G + q_H + q_D + q_G q_D) + q_D \Big(T[f] + T[\mathsf{Dec}^{sym}] \Big) + T$$

where $T[f]$ is the time complexity of f.

Proof: It suffices to invoke the PCA oracle into the plaintext checking algorithm $\mathcal{V}$ for f. Thus, by definition of oracle access, $T[\mathcal{V}] = 1$. ■

5.4 Particular cases

Both in the case of the trivial construction of partial one-way bijection families and in the non-trivial family defined in subsection 3.1, the simulation in the security proof can be improved introducing only technical modifications.

In both cases, there exist a polynomial size set family Z' and two very efficiently computable function families $f' : X \rightarrow Z'$ and $\pi' : \bar{Z} \rightarrow Z'$ such that for all $pk \in PK$, $x \in X_{pk}$ and $z \in \bar{Z}_{pk}$, $\mathcal{V}(pk, x, z) = 1$ if and only if $f'_{pk}(x) = \pi'_{pk}(z)$. Notice that this property implies the injectivity of f'. It is shown in the appendix that

$$\begin{aligned} T[\mathcal{A}_{\mathsf{POW}(f)}] &\le (q_G + q_H + q_D)T[\mathcal{V}] + q_G T[f'] + \\ &\quad + q_D\Big(T[f] + T[\pi'] + T[\mathsf{Dec}^{sym}]\Big) + T[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}] \end{aligned}$$

then providing a very-tight security reduction.

If the trivial constructions are considered, $f_{pk}(x, y) = (f'_{pk}(x), y)$ and $\pi'_{pk}(z', y) = z'$. Then, $T[\pi']$ can be neglected and $T[f] \approx T[f'] \approx T[\mathcal{V}]$. So

$$T[\mathcal{A}_{\mathsf{POW}(f)}] \le (2q_G + q_H + 2q_D)T[f'] + q_D T[\mathsf{Dec}^{sym}] + T[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}]$$

On the other hand, if the generalised RSA-Paillier function is used, $f'_{n,r,e}(x) = x^e \bmod n$ and $\pi'_{n,r,e}(z) = z \bmod n$. Then,

$$T[\mathcal{A}_{\mathsf{POW}(f)}] \le (2q_G + q_H + 2q_D)O(\ell^2 \log e) + q_D T[\mathsf{Dec}^{sym}] + T[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}]$$

References

[1] M. Bellare, A. Desai, D. Pointcheval and P. Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. *CRYPTO '98, LNCS* **1462** 26–45 (1998).

[2] M. Bellare and P. Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. *ACM CCS 93, ACM Press* (1993)

[3] R. Canetti, O. Goldreich and S. Halevi. The random oracle methodology, revisited. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, 209–218 (1998).

[4] D. Catalano, R. Gennaro, N. Howgrave-Graham and P. Q. Nguyen. Paillier's Cryptosystem Revisited. *ACM CCS '2001 ACM Press* (2001).

[5] J. Coron, H. Handschuh, M. Joye, P. Paillier, D. and C. Tymen. GEM: a Generic Chosen-Ciphertext Secure Encryption Method. *CT-RSA' 02, LNCS* **2271** 263–276 (2002).

[6] J. Coron, H. Handschuh, M. Joye, P. Paillier, D. Pointcheval and C. Tymen. Optimal Chosen-Ciphertext Secure Encryption of Arbitrary-Length Messages. *PKC 2002, LNCS* **2274** 17–33 (2002).

[7] R. Cramer and V. Shoup. Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. *EUROCRYPT '2002, LNCS* **2332** 45–64 (2002).

[8] A. W. Dent. An implementation attack against the EPOC-2 public-key cryptosystem. *Electronics Letters* VOL. 38 NO. 9 412–413 (2002).

[9] EPOC, Efficient Probabilistic Public-Key Encryption. `http://info.isl.ntt.co.jp/epoc/`

[10] E. Fujisaki and T. Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. *CRYPTO '99, LNCS* **1666** 537–554 (1999)

[11] E. Fujisaki and T. Okamoto. A Chosen-Cipher Secure Encryption Scheme Tightly as Secure as Factoring. *IEICE Trans. Fundamentals* **E84-A**(1) 179–187 (2001).

[12] M. Joye, J. J. Quisquater and M. Yung. On the Power of Misbehaving Adversaries and Security Analysis of the Original EPOC. *CT-RSA' 01, LNCS* **2020** 208–222 (2001).

[13] T. Okamoto and D. Pointcheval. The Gap-Problems: a New Class of Problems for the Security of Cryptographic Schemes. *PKC' 01, LNCS* **1992** 104–118 (2001).

[14] T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. *CT-RSA' 01, LNCS* **2020** 159–175 (2001).

[15] T. Okamoto and S. Uchiyama. A New Public-Key Cryptosystem as Secure as Factoring. *EUROCRYPT-98, LNCS* **1403** 308–318 (1998)

[16] PSEC, Provably Secure Encryption Scheme. `http://info.isl.ntt.co.jp/psec/`

[17] D. Pointcheval. Chosen-Ciphertext Security for any One-Way Cryptosystem. *Proc. PKC '2000 LNCS* **1751** 129–146 (2000).

[18] K. Sakurai and T. Takagi. A Reject Timing Attack on an IND-CCA2 Public-Key Cryptosystem. *ICISC '02, LNCS* **2587** 359–373 (2002).

[19] Y. Watanabe, J. Shikata and H.Imai. Equivalence between Semantic Security and Indistinguishability against Chosen Ciphertext Attacks. *PKC 2003, LNCS* **2567** 71–84 (2002).

A Proof of theorem 9

Let $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}[T, \epsilon, q_G, q_H, q_D] = (\mathcal{A}_1, \mathcal{A}_2)$ be the adversary aiming to attack the IND-CCA security of the hybrid encryption scheme, $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ described in subsection 5.3.

In order to prove the theorem, some different games will be considered. In all games, the adversary $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ uses the same coins, but the events defined as functions of the view of $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ could occur with different probabilities in each game. Starting from the IND-CCA game we will design several intermediate games before designing the game for an adversary who tries to break the partial one-wayness (POW) of f. Each game will be obtained by introducing slight modifications to the previous game in such a way that the adversary success probabilities are easily related. We denote by $\mathsf{Pr}_i[F]$ the probability of event F in game i.

Each game will be described as a main algorithm along with some auxiliar algorithms used as oracles by $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$. The bulleted steps in the algorithms will indicate the main changes introduced in each game.

The following trivial lemma will be very useful in this proof.

Lemma 11 *Let E_1, F_1 be two events defined in a probability space $\mathcal{X}_1$, and E_2, F_2 another two events defined in a probability space $\mathcal{X}_2$, such that $p = \mathsf{Pr}_{\mathcal{X}_2}[F_2] = \mathsf{Pr}_{\mathcal{X}_1}[F_1]$ and $\mathsf{Pr}_{\mathcal{X}_2}[E_2 \wedge \neg F_2] = \mathsf{Pr}_{\mathcal{X}_1}[E_1 \wedge \neg F_1]$. Then*

$$|\mathsf{Pr}_{\mathcal{X}_2}[E_2] - \mathsf{Pr}_{\mathcal{X}_1}[E_1]| \leq p$$

Game0. The IND-CCA attack. There are some minor differences between Game0 and the standard IND-CCA game, described in subsection 4.2, but they do not modify any probability.

```
Game0()
  1  $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell);\ G \leftarrow \mathcal{R}(K_\ell);\ H \leftarrow \mathcal{R}(Y_{pk})$
  2  $b \leftarrow \{0,1\};\ x^\star \leftarrow X_{pk}$
  3  $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{G,H,\mathcal{D}_{sk}}(pk)$
  4  $y^\star \leftarrow H(x^\star, m_b);\ c^\star \leftarrow \Big(f_{pk}(x^\star, y^\star), \mathsf{Enc}^{sym}_{G(x^\star)}(m_b)\Big)$
  5  $b' \leftarrow \mathcal{A}_2^{G,H,\mathcal{D}_{sk,c^\star}}(s, c^\star)$
```

where the oracle answer $\mathcal{D}_{sk}(c)$ is exactly the same as the value returned by $\mathsf{Dec}(sk, c)$, described in subsection 5.3.

Let Askx be the event that, during the game, either $x^\star \in X$ is queried (by $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$) to G or $(x^\star, m)$ is queried to H, for some m. Then,

$$\begin{aligned}\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}\right] &= |\mathsf{Pr}_0[b' = b] - \mathsf{Pr}_0[b' \neq b]| \leq \\ &\leq |\mathsf{Pr}_0[b' = b \wedge \mathsf{Askx}] - \mathsf{Pr}_0[b' \neq b \wedge \mathsf{Askx}]| + \\ &\quad + |\mathsf{Pr}_0[b' = b \wedge \neg\mathsf{Askx}] - \mathsf{Pr}_0[b' \neq b \wedge \neg\mathsf{Askx}]| \leq \\ &\leq \mathsf{Pr}_0[\mathsf{Askx}] + |\mathsf{Pr}_0[b' = b \wedge \neg\mathsf{Askx}] - \mathsf{Pr}_0[b' \neq b \wedge \neg\mathsf{Askx}]|\end{aligned}$$

In order to improve the readability of the rest of the proof, let define $\mathsf{S}_1 = \mathsf{Askx}$, $\mathsf{S}_{01} = \neg\mathsf{Askx} \wedge b' = b$ and $\mathsf{S}_{00} = \neg\mathsf{Askx} \wedge b' \neq b$. The above equation can be rewritten as

$$\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}\right] \leq \mathsf{Pr}_0[\mathsf{S}_1] + |\mathsf{Pr}_0[\mathsf{S}_{01}] - \mathsf{Pr}_0[\mathsf{S}_{00}]|$$

Let $\mathcal{T}_G$ be a table in which all the queries made by $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ to the oracle G are stored along with the corresponding answers. Notice that $\mathcal{T}_G$ would not contain ALL the queries made to G. In the sequel, $x \in \mathcal{T}_G$ will denote the fact that x has been queried to G by $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ and $\mathcal{T}_G(x)$ will denote the answer given by G. Define $(x, m) \in \mathcal{T}_H$, $\mathcal{T}_H(x, m)$ and $c \in \mathcal{T}_{\mathcal{D}}$, $\mathcal{T}_{\mathcal{D}}(c)$ in a similar way respectively for the oracle calls to H and $\mathcal{D}_{sk}$. Notice that the contents of these tables will vary during the game.

Game1. In this game, the queries made by $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ to the two random oracles are intercepted in order to immediately abort the execution of the game if Askx (i.e. S_1) occurs. The following functions will do this task:

```
$G1(x)$
1  if $x = x^\star$; exit Game; endif
2  return $G(x)$
```

```
$H1(x, m)$
1  if $x = x^\star$; exit Game; endif
2  return $H(x, m)$
```

and the new game is,

```
Game1()
1  $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell)$; $G \leftarrow \mathcal{R}(K_\ell)$; $H \leftarrow \mathcal{R}(Y_{pk})$
2  $b \leftarrow \{0,1\}$; $b' \leftarrow \{0,1\}$; $x^\star \leftarrow X_{pk}$
3  $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{G1,H1,\mathcal{D}_{sk}}(pk)$
4  $y^\star \leftarrow H(x^\star, m_b)$; $c^\star \leftarrow \left(f_{pk}(x^\star, y^\star), \mathsf{Enc}^{sym}_{G(x^\star)}(m_b)\right)$
5  $b' \leftarrow \mathcal{A}_2^{G1,H1,\mathcal{D}_{sk,c^\star}}(s, c^\star)$
```

Since the games are identical while $\neg\mathsf{S}_1$, the events S_1, S_{01} and S_{00} remain unchanged in Game1. Then,

$$\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}\right] \leq \mathsf{Pr}_1[\mathsf{S}_1] + |\mathsf{Pr}_1[\mathsf{S}_{01}] - \mathsf{Pr}_1[\mathsf{S}_{00}]|$$

Game2. In this game, the decryption oracle is modified in such a way that it is disallowed to do new queries to the random oracle G. To do this, all ciphertexts (c_1, c_2) submitted to the decryption oracle such that $g_{sk}(c_1) \notin \mathcal{T}_G \cap X_{pk}$ are rejected by returning $\bot_2$, even when some of them may be valid ciphertexts.

```
Game2()
1   $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell);\ G \leftarrow \mathcal{R}(K_\ell);\ H \leftarrow \mathcal{R}(Y_{pk})$
2   $b \leftarrow \{0,1\};\ x^\star \leftarrow X_{pk}$
3   $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{G1,H1,\mathcal{D}2_{sk}}(pk)$
4   $y^\star \leftarrow H(x^\star, m_b);\ c^\star \leftarrow \left(f_{pk}(x^\star, y^\star), \mathsf{Enc}^{sym}_{G(x^\star)}(m_b)\right)$
5   $b' \leftarrow \mathcal{A}_2^{G1,H1,\mathcal{D}2_{sk,c^\star}}(s, c^\star)$
```

```
$\mathcal{D}2_{sk}(c)$
  1   if $c \notin \bar{Z}_{pk} \times M_\ell$; return $\bot_1$; endif
  2   $(c_1, c_2) = c$
  3   $x \leftarrow g_{sk}(c_1)$
• 4   if $x \notin X_{pk}$ or $x \notin \mathcal{T}_G$; return $\bot_2$; endif
  5   $m \leftarrow \mathsf{Dec}^{sym}_{G(x)}(c_2)$
  6   $y \leftarrow H(x, m)$
  7   if $f_{pk}(x, y) \neq c_1$; return $\bot_2$; endif
  8   return $m$
```

Let F_2 be the event that, in some query to the decryption oracle, the ciphertext is accepted in Game1, but is rejected at step 4 of $\mathcal{D}2_{sk}$. Before F_2, both games are indentical. Then, by lemma 11,

$$\begin{aligned}|\mathsf{Pr}_2[\mathsf{S}_1] - \mathsf{Pr}_1[\mathsf{S}_1]| &\leq \mathsf{Pr}[F_2]\\ |\mathsf{Pr}_2[\mathsf{S}_{01}] - \mathsf{Pr}_1[\mathsf{S}_{01}]| &\leq \mathsf{Pr}[F_2]\\ |\mathsf{Pr}_2[\mathsf{S}_{00}] - \mathsf{Pr}_1[\mathsf{S}_{00}]| &\leq \mathsf{Pr}[F_2]\end{aligned}$$

From these inequalities, it can be easily shown that

$$\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}\right] \leq \mathsf{Pr}_2[\mathsf{S}_1] + |\mathsf{Pr}_2[\mathsf{S}_{01}] - \mathsf{Pr}_2[\mathsf{S}_{00}]| + 2\mathsf{Pr}[F_2]$$

The following lemma gives an upper bound for $\mathsf{Pr}[F_2]$.

Lemma 12

$$\mathsf{Pr}[F_2] \leq \frac{q_D q_H \gamma}{|K| - q_D q_H \gamma} + \frac{q_D}{|Y| - q_D}$$

Proof: Let Fail_k be the event that F_2 occurs exactly at the k-th query to the decryption oracle. Let $\mathsf{NoFail}_k = \wedge_{j=1}^k \neg\mathsf{Fail}_j$ denote the event that F_2 does not occur during the first k queries to the decryption oracle, for $k = 1, \ldots, q_D$. Let NoFail_0 be the certain event. Then,

$$\mathsf{Pr}[F_2] = 1 - \mathsf{Pr}[\mathsf{NoFail}_{q_D}] = 1 - \mathsf{Pr}\left[\bigwedge_{k=1}^{q_D} \neg\mathsf{Fail}_k\right] = 1 - \prod_{k=1}^{q_D} \mathsf{Pr}[\neg\mathsf{Fail}_k \mid \mathsf{NoFail}_{k-1}]$$

and, if we denote by $p_k = \mathsf{Pr}\left[\mathsf{Fail}_k \mid \mathsf{NoFail}_{k-1}\right]$, then

$$\mathsf{Pr}\left[F_2\right] = 1 - \prod_{k=1}^{q_D}(1-p_k) \leq \sum_{k=1}^{q_D} p_k$$

In order to compute p_k, let us suppose that Game1 and Game2 run identically just until $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ is going to submit the k-th query, $\bar{c}$, to the decryption oracle. This implies $\mathsf{NoFail}_{k-1} \wedge \neg\mathsf{Askx}$. Suppose for a while that $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ is in the 'finding' stage. The only information available to the adversary, in order to generate the cyphertext $\bar{c}$ is the view of the game at this execution point, that is $\mathsf{View} = (pk, \mathcal{T}_G, \mathcal{T}_H, \mathcal{T}_\mathcal{D})$. To find an upper bound for p_k, we will consider the best choice for $\bar{c}$, for each possible View compatible with NoFail_{k-1}.

The event Fail_k occurs if and only if $\mathcal{D}2_{sk}(\bar{c}) \neq \mathcal{D}_{sk}(\bar{c})$, that is, $\mathcal{D}2_{sk}$ rejects $\bar{c}$ (returning $\perp_2$) while $\mathcal{D}_{sk}$ accepts it. This means that $\bar{c} = (f_{pk}(\bar{x}, \bar{y})), \bar{c}_2)$, where $\bar{x} \in X_{pk} \setminus \mathcal{T}_G$, $\bar{y} \in Y_{pk}$, $\bar{c}_2 \in M_\ell$, and the equation $\bar{y} = H(\bar{x}, \mathsf{Dec}^{sym}_{G(\bar{x})}(\bar{c}_2))$ holds.

If View and $\bar{c}$ are fixed, then p_k depends only on the joint probability distribution of $G(\bar{x})$ and $H(\bar{x}, \mathsf{Dec}^{sym}_{G(\bar{x})}(\bar{c}_2))$. But this distribution is conditioned by the answers given by H to the queries $(\bar{x}, m)$ for some m, and the answers given by $\mathcal{D}_{sk}$ to the queries $(f_{pk}(\bar{x}, y), c_2)$ for some $y \in Y_{pk}$ and $c_2 \in M_\ell$. Notice that any queried ciphertext $c \notin Z_{pk} \times M_\ell$ is rejected by $\mathcal{D}_{sk}$, independently of the values taken by the random functions.

In the worst case, all queries in $\mathcal{T}_H$ and $\mathcal{T}_\mathcal{D}$ are related to $\bar{x}$, that is, $h_i = H(\bar{x}, m_i)$ for $i = 1, \ldots, q_H$, and $c^{(j)} = (f_{pk}(\bar{x}, y_j), c_2^{(j)})$ for $j = 1, \ldots, k-1$. Since $\bar{x} \notin \mathcal{T}_G$, then $\mathcal{D}_{sk}(c^{(j)}) = \mathcal{D}2_{sk}(c^{(j)}) = \perp_2$ and then $y_j \neq H(\bar{x}, \mathsf{Dec}^{sym}_{G(\bar{x})}(c_2^{(j)}))$. These equations could be incompatible for some values of $G(\bar{x})$, namely those $g \in K_\ell$ such that $m_i = \mathsf{Dec}^{sym}_g(c_2^{(j)})$ and $h_i = y_j$ for some (i,j). In the (unfeasible) worst case, all h_i and y_j are equal and there can be up to $q_H(k-1)\gamma$ forbidden values for $G(\bar{x})$. Then, the random variable $G(\bar{x})$ is uniformly distributed over a set of at least $|K_\ell| - (k-1)q_H\gamma$ elements.

There are at most $q_H\gamma$ different values of g such that $(\bar{x}, \mathsf{Dec}^{sym}_g(\bar{c}_2)) \in \mathcal{T}_H$. For these values, $\bar{y} = H(\bar{x}, \mathsf{Dec}^{sym}_g(\bar{c}_2))$ can be ensured if all h_i are equal to $\bar{y}$. Thus,

$$\mathsf{Pr}\left[\mathsf{Fail}_k \wedge (\bar{x}, \mathsf{Dec}^{sym}_{G(\bar{x})}(\bar{c}_2)) \in \mathcal{T}_H \mid \mathsf{View}\right] \leq \frac{q_H\gamma}{|K_\ell| - (k-1)q_H\gamma}$$

For any g such that $(\bar{x}, \mathsf{Dec}^{sym}_g(\bar{c}_2)) \notin \mathcal{T}_H$, the variable $H(\bar{x}, \mathsf{Dec}^{sym}_g(\bar{c}_2))$ is uniformly distributed over a set of at least $|Y_{pk}| - (k-1)$ elements, because if $\bar{c}_2 = c_2^{(j)}$, then the value y_j is forbidden. Consequently,

$$\mathsf{Pr}\left[\mathsf{Fail}_k \wedge (\bar{x}, \mathsf{Dec}^{sym}_{G(x)}(c_2)) \notin \mathcal{T}_H \mid \mathsf{View}\right] \leq \frac{1}{|Y_{pk}| - (k-1)}$$

and summing up, we obtain

$$p_k^{\text{find}} \leq \frac{q_H\gamma}{|K_\ell| - (k-1)q_H\gamma} + \frac{1}{|Y_{pk}| - (k-1)}$$

If $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ is in the 'guessing' stage, then $c^\star$ holds valuable information. In fact, $\mathsf{View} = (pk, \mathcal{T}_G, \mathcal{T}_H, \mathcal{T}_\mathcal{D}, c^\star)$, but $c^\star$ depends only on $G(x^\star)$ and $H(x^\star, m_b)$. Thus, if $\bar{x} \neq x^\star$, $c^\star$ does not give any additional information about Fail_k and everithing goes the same way as in the 'finding' stage.

If $\bar{x} = x^\star$, also the restriction $\bar{c} \neq c^\star$ must be considered. Moreover, there are no queries in $\mathcal{T}_H$ related to $x^\star$. Then, in the worst case, the joint distribution of $G(\bar{x})$ and $H(\bar{x}, \mathsf{Dec}^{sym}_{G(\bar{x})}(\bar{c}_2))$ is conditioned by the equations $y_j \neq H(x^\star, \mathsf{Dec}^{sym}_{G(x^\star)}(c_2^{(j)}))$, for $j = 1\ldots, k-1$, $y^\star = H(x^\star, m_b)$ and $m_b = \mathsf{Dec}^{sym}_{G(x^\star)}(c_2^\star)$.

The equality $y^\star = H(x^\star, m_b)$ is useless since the only valid ciphertext related to $H(x^\star, m_b)$ is $c^\star$. Nevertheless, from $m_b = \mathsf{Dec}^{sym}_{G(x^\star)}(c_2^\star)$, only a reduced number of values of $G(x^\star)$ remain possible, but, as above, $H(x^\star, \mathsf{Dec}^{sym}_{G(x^\star)}(\bar{c}_2))$ is uniformly distributed over a set of at least $|Y_{pk}| - (k-1)$ elements, and $p_k^{\text{guess}} \leq \dfrac{1}{|Y_{pk}| - (k-1)}$.

Finally,

$$\mathsf{Pr}\,[F_2] \leq \sum_{k=1}^{q_D} \left(\frac{q_H \gamma}{|K_\ell| - (k-1) q_H \gamma} + \frac{1}{|Y_{pk}| - (k-1)} \right) \leq \frac{q_D q_H \gamma}{|K| - q_D q_H \gamma} + \frac{q_D}{|Y| - q_D}$$

■

Game2′. In this game, oracles G and H are simulated by using tables $\mathcal{T}_{G2'}$ and $\mathcal{T}_{H2'}$, as described in subsection 4.3.

Also, the generation of the ciphertext differs from the one in Game2. Here, in Game2′, some values of the random functions are redefined, namely $G(x^\star) = g$ and $H(x^\star, m_b) = y^\star$. But these changes in the oracles do not affect the probability distribution of the view of $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$, since in Game2 neither $x^\star$ is queried to G nor $(x^\star, m)$ is queried to H, for any m. (Note that, at step 6 of $\mathcal{D}2_{sk}$, $x \neq x^\star$ since $x^\star \notin \mathcal{T}_G$.)

```
Game2′()
• 1  $\mathcal{T}_{G2'} \leftarrow$ empty ; $\mathcal{T}_{H2'} \leftarrow$ empty
  2  $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell)$
  3  $b \leftarrow \{0,1\}$;  $x^\star \leftarrow X_{pk}$
  4  $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{G2', H2', \mathcal{D}2'_{sk}}(pk)$
• 5  $g \leftarrow K_\ell$;  $y^\star \leftarrow Y_{pk}$;  $c^\star \leftarrow \big(f_{pk}(x^\star, y^\star), \mathsf{Enc}^{sym}_g(m_b)\big)$
  6  $b' \leftarrow \mathcal{A}_2^{G2', H2', \mathcal{D}2'_{sk,c^\star}}(s, c^\star)$
```

```
$\mathcal{D}2'_{sk}(c)$
  1  if $c \notin \bar{Z}_{pk} \times M_\ell$;  return $\perp_1$;  endif
  2  $(c_1, c_2) = c$
  3  $x \leftarrow g_{sk}(c_1)$
• 4  if $x \notin X_{pk}$ or $x \notin \mathcal{T}_{G2'}$;  return $\perp_2$;  endif
  5  $m \leftarrow \mathsf{Dec}^{sym}_{G2'(x)}(c_2)$
  6  $y \leftarrow H2'(x, m)$
  7  if $f_{pk}(x, y) \neq c_1$;  return $\perp_2$;  endif
  8  return $m$
```

$G2'(x)$

```
• 1  if $x \in \mathcal{T}_{G2'}$; return $\mathcal{T}_{G2'}(x)$; endif
  2  if $x = x^\star$; exit Game; endif
• 3  $r \leftarrow K_\ell$
• 4  insert $(x, r)$ in table $\mathcal{T}_{G2'}$
  5  return $r$
```

$H2'(x, m)$

```
• 1  if $(x, m) \in \mathcal{T}_{H2'}$; return $\mathcal{T}_{H2'}(x, m)$; endif
  2  if $x = x^\star$; exit Game; endif
• 3  $r \leftarrow Y_{pk}$
• 4  insert $((x, m), r)$ in table $\mathcal{T}_{H2'}$
  5  return $r$
```

Game3. In this game, we introduce some modifications to avoid the use of m_b in the generation of the target ciphertext. In fact, the differences between using m_b and using a random message can be tapped by a new adversary $\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})} = (\mathcal{A}_1^{sym}, \mathcal{A}_2^{sym})$ who tries to break the IND-SYM security of $\mathcal{E}^{sym}$ (see 4.1).

Game3()

```
1  $\beta \leftarrow \{0, 1\}$
2  $(\mu_0, \mu_1, \sigma) \leftarrow \mathcal{A}_1^{sym}(1^\ell)$
3  $g \leftarrow K_\ell$; $\kappa^\star = \mathsf{Enc}_g^{sym}(\mu_\beta)$
4  $\beta' \leftarrow \mathcal{A}_2^{sym}(\sigma, \kappa^\star)$
```

$\mathcal{A}_1^{sym}(1^\ell)$

```
1  $\mathcal{T}_{G3} \leftarrow$ empty ; $\mathcal{T}_{H3} \leftarrow$ empty
2  $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell)$
3  $b \leftarrow \{0, 1\}$; $x^\star \leftarrow X_{pk}$
4  $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{G3, H3, \mathcal{D}3_{sk}}(pk)$
5  $m \leftarrow M_\ell$
6  $\sigma = (\mathcal{T}_{G3}, \mathcal{T}_{H3}, pk, sk, b, x^\star, s)$
7  return $(m_b, m, \sigma)$
```

$\mathcal{A}_2^{sym}(\sigma, \kappa^\star)$

```
1  $(\mathcal{T}_{G3}, \mathcal{T}_{H3}, pk, sk, b, x^\star, s) = \sigma$
2  $y^\star \leftarrow Y_{pk}$; $c^\star \leftarrow (f_{pk}(x^\star, y^\star), \kappa^\star)$
3  $b' \leftarrow \mathcal{A}_2^{G3, H3, \mathcal{D}3_{sk, c^\star}}(s, c^\star)$
4  $\beta'' \leftarrow 0$
5  if $b' = b$
6      $\beta' \leftarrow 0$
7  else
8      $\beta' \leftarrow 1$
9  endif
```

$\mathcal{D}3_{sk}(c)$

```
  if $c \notin \bar{Z}_{pk} \times M_\ell$; return $\perp_1$; endif
  $(c_1, c_2) = c$
  $x \leftarrow g_{sk}(c_1)$
• if $x \notin X_{pk}$ or $x \notin \mathcal{T}_{G3}$; return $\perp_2$; endif
  $m \leftarrow \mathsf{Dec}^{sym}_{G3(x)}(c_2)$
  $y \leftarrow H3(x, m)$
  if $f_{pk}(x, y) \neq c_1$; return $\perp_2$; endif
  return $m$
```

$G3(x)$

```
  if $x \in \mathcal{T}_{G3}$; return $\mathcal{T}_{G3}(x)$; endif
  if $x = x^\star$
•     $\beta' \leftarrow \{0, 1\}$
•     $\beta'' \leftarrow 1$
      exit Game
  endif
  $r \leftarrow K_\ell$
  insert $(x, r)$ in table $\mathcal{T}_{G3}$
  return $r$
```

$H3(x, m)$

```
• if $(x, m) \in \mathcal{T}_{H3}$; return $\mathcal{T}_{H3}(x, m)$; endif
  if $x = x^\star$
•     $\beta' \leftarrow \{0, 1\}$
•     $\beta'' \leftarrow 1$
      exit Game
  endif
  $r \leftarrow Y_{pk}$
  insert $((x, m), r)$ in table $\mathcal{T}_{H3}$
  return $r$
```

Actually, $\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}$ uses two different ways to guess the value of β: β' indicates if $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ guesses the correct value of b and β'' indicates if S_1 occurs. Then, two different advantages can be taken into account: $\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right] = |2\mathsf{Pr}_3\left[\beta' = \beta\right] - 1|$ and $\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right]' = |2\mathsf{Pr}_3\left[\beta'' = \beta\right] - 1|$.

If $\beta = 1$, the value of m_b is used nowhere in the game. So, the view of $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ is independent of b and $\mathsf{Pr}_3\left[\beta' = 1 \mid \beta = 1 \wedge \neg\mathsf{S}_1\right] = \mathsf{Pr}_3\left[b' \neq b \mid \beta = 1 \wedge \neg\mathsf{S}_1\right] = \frac{1}{2}$. Moreover, $\mathsf{Pr}_3\left[\beta' = 1 \mid \beta = 1 \wedge \mathsf{S}_1\right] = \frac{1}{2}$ and then $\mathsf{Pr}_3\left[\beta' = 1 \mid \beta = 1\right] = \frac{1}{2}$.

If $\beta = 0$, Game3 and Game2$'$ are identical. Thus

$$\mathsf{Pr}_3\left[\beta' = 0 \wedge \neg\mathsf{S}_1 \mid \beta = 0\right] = \mathsf{Pr}_3\left[b' = b \wedge \neg\mathsf{S}_1 \mid \beta = 0\right] = \mathsf{Pr}_2\left[\mathsf{S}_{01}\right]$$

and

$$\mathsf{Pr}_3\left[\beta' = 0 \wedge \mathsf{S}_1 \mid \beta = 0\right] = \frac{1}{2}\mathsf{Pr}_3\left[\mathsf{S}_1 \mid \beta = 0\right] = \frac{1}{2}\mathsf{Pr}_2\left[\mathsf{S}_1\right]$$

Putting altogether,

$$\begin{aligned}\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right] &= \left|2\mathsf{Pr}_3\left[\beta' = 0 \wedge \beta = 0\right] + 2\mathsf{Pr}_3\left[\beta' = 1 \wedge \beta = 1\right] - 1\right| = \\ &= \left|\mathsf{Pr}_3\left[\beta' = 0 \mid \beta = 0\right] + \mathsf{Pr}_3\left[\beta' = 1 \mid \beta = 1\right] - 1\right| = \\ &= \left|\mathsf{Pr}_2\left[\mathsf{S}_{01}\right] + \tfrac{1}{2}\mathsf{Pr}_2\left[\mathsf{S}_1\right] - \tfrac{1}{2}\right| = \tfrac{1}{2}\left|\mathsf{Pr}_2\left[\mathsf{S}_{01}\right] - \mathsf{Pr}_2\left[\mathsf{S}_{00}\right]\right|\end{aligned}$$

If β'' is used instead of β', then

$$\begin{aligned}\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right]' &= \left|2\mathsf{Pr}_3\left[\mathsf{S}_1 \wedge \beta'' = \beta\right] + 2\mathsf{Pr}_3\left[\neg\mathsf{S}_1 \wedge \beta'' = \beta\right] - 1\right| = \\ &= \left|2\mathsf{Pr}_3\left[\mathsf{S}_1 \wedge \beta = 1\right] + 2\mathsf{Pr}_3\left[\neg\mathsf{S}_1 \wedge \beta = 0\right] - 1\right| = \\ &= \left|\mathsf{Pr}_3\left[\mathsf{S}_1 \mid \beta = 1\right] + \left(\mathsf{Pr}_3\left[\neg\mathsf{S}_1 \mid \beta = 0\right] - 1\right)\right| = \\ &= \left|\mathsf{Pr}_3\left[\mathsf{S}_1 \mid \beta = 1\right] - \mathsf{Pr}_2\left[\mathsf{S}_1\right]\right|\end{aligned}$$

Finally,

$$\begin{aligned}\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}\right] &\leq \mathsf{Pr}_2\left[\mathsf{S}_1\right] + \left|\mathsf{Pr}_2\left[\mathsf{S}_{01}\right] - \mathsf{Pr}_2\left[\mathsf{S}_{00}\right]\right| + 2\mathsf{Pr}\left[F_2\right] = \\ &= \mathsf{Pr}_2\left[\mathsf{S}_1\right] + 2\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right] + 2\mathsf{Pr}\left[F_2\right] \leq \\ &\leq \mathsf{Pr}_3\left[\mathsf{S}_1 \mid \beta = 1\right] + 2\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right] + \\ &\quad + \mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right]' + 2\mathsf{Pr}\left[F_2\right]\end{aligned}$$

Game4. Game3 (with $\beta = 0$) can be modified to obtain an implementation of an adversary, $\mathcal{A}_{\mathsf{POW}(f)}$, that try to break the partial one-wayness of f. This adversary will know neither sk nor $x^\star$. The use of sk in the decryption oracle simulator and the use of $x^\star$ in the random oracle simulators are avoided conveniently using the deterministic plaintext checking algorithm $\mathcal{V}$. The value of $x^\star$ is guessed by $\mathcal{A}_{\mathsf{POW}(f)}$ when possible (i.e. if S_1 occurs).

These changes do not modify any probability. Moreover, the views of $\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}$ in games 3 (with $\beta = 0$) and 4 are identically distributed.

```
Game4()
    $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell)$
    $x^\star \leftarrow X_{pk};\ y^\star \leftarrow Y_{pk};\ z \leftarrow f_{pk}(x^\star, y^\star)$
    $\mathcal{A}_{\mathsf{POW}(f)}(pk, z^\star)$

$\mathcal{A}_{\mathsf{POW}(f)}(pk, z)$
    $b \leftarrow \{0, 1\}$
    $m \leftarrow M_\ell;\ g \leftarrow K_\ell;\ c^\star \leftarrow (z, \mathsf{Enc}^{sym}_g(m))$
    $\mathcal{T}_{G4} \leftarrow$ empty ; $\mathcal{T}_{H4} \leftarrow$ empty
    $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{G4, H4, \mathcal{D}4_{pk}}(pk)$
    $b' \leftarrow \mathcal{A}_2^{G4, H4, \mathcal{D}4_{pk,c^\star}}(s, c^\star)$
•   $x' \leftarrow X_{pk}$

$G4(x)$
    if $x \in \mathcal{T}_{G4}$; return $\mathcal{T}_{G4}(x)$; endif
•   if $x \in X_{pk}$ and $\mathcal{V}(pk, x, z) = 1$
•       $x' \leftarrow x$
        exitGame
    endif
    $r \leftarrow K_\ell$
```

```
  insert $(x, r)$ in table $\mathcal{T}_{G4}$
  return $r$
```

$H4(x, m)$

```
  if $(x, m) \in \mathcal{T}_{H4}$; return $\mathcal{T}_{H4}(x, m)$; endif
● if $x \in X_{pk}$ and $\mathcal{V}(pk, x, z) = 1$
●   $x' \leftarrow x$
    exitGame
  endif
  $r \leftarrow Y_{pk}$
  insert $((x, m), r)$ in table $\mathcal{T}_{H4}$
  return $r$
```

$\mathcal{D}4_{pk}(c)$

```
  if $c \notin \bar{Z}_{pk} \times M_\ell$; return $\perp_1$; endif
  $(c_1, c_2) = c$
● foreach $x \in \mathcal{T}_{G4}$
●   if $x \in X_{pk}$ and $\mathcal{V}(pk, x, c_1) = 1$
      $m \leftarrow \mathsf{Dec}^{sym}_{\mathcal{T}_{G4}(x)}(c_2)$
      $y \leftarrow H4(x, m)$
      if $f_{pk}(x, y) \neq c_1$; return $\perp_2$; endif
      return $m$
    endif
● endforeach
  return $\perp_2$
```

Now,

$$\mathsf{Succ}\left[\mathcal{A}_{\mathsf{POW}(f)}\right] = \mathsf{Pr}_4\left[x' = x^\star\right] \geq \mathsf{Pr}_4\left[\mathsf{S}_1\right] = \mathsf{Pr}_3\left[\mathsf{S}_1 \mid \beta = 1\right]$$

and, from the above results,

$$\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}\right] \leq \mathsf{Succ}\left[\mathcal{A}_{\mathsf{POW}(f)}\right] + 2\mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right] + \\ + \mathsf{Adv}\left[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}\right]' + \frac{2q_D q_H \gamma}{|K| - q_D q_H \gamma} + \frac{2q_D}{|Y| - q_D}$$

In terms of time complexity of the algorithms, the overhead introduced by the simulation of the random oracles, G and H, in games 3 and 4 can be reduced by using standard hashing techniques for table insertion and searching. In fact, in almost all security proofs in the Random Oracle Model in the literature, this time overhead is neglected. It is also supposed that the times needed to check if $c \in \bar{Z}_{pk} \times M_\ell$ and $x \in X_{pk}$ are negligible.

Neglecting lower order terms, the running time of $\mathcal{A}_{\mathsf{POW}(f)}$ in Game4 is bounded by

$$T[\mathcal{A}_{\mathsf{POW}(f)}] \leq (q_G + q_H + q_D + q_G q_D)T[\mathcal{V}] + q_D\Big(T[f] + T[\mathsf{Dec}^{sym}]\Big) + T[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}]$$

where $T[\mathcal{V}]$ is the time complexity of the plaintext checking algoritm and $T[f]$ is the time complexity of f. Also, $T[\mathcal{A}_{\mathsf{IND-SYM}(\mathcal{E}^{sym})}] = T[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}]$.

A.1 Particular cases

Both in the case of the trivial construction of easy verifiable functions and in the non-trivial family in subsection 3.1, the algorithm $\mathcal{D}4_{pk}$ can be improved, without modifying the behavior of the game, to avoid exhaustive search in $\mathcal{T}_{G4}$. To do it, $(f'(x),(x,G(x)))$ is stored in another table $\mathcal{T}'_{G4}$ for each query $x \in X_{pk}$ to G.

```
$G4'(x)$
   1   if $x \in \mathcal{T}_{G4}$; return $\mathcal{T}_{G4}(x)$; endif
•  2   if $x \in X_{pk}$ and $\mathcal{V}(pk,x,z) = 1$
•  3       $x' \leftarrow x$
   4       exitGame
   5   endif
   6   $r \leftarrow K_\ell$
   7   insert $(x,r)$ in table $\mathcal{T}_{G4}$
•  8   if $x \in X_{pk}$
•  9       insert $(f'(x),(x,r))$ in table $\mathcal{T}'_{G4}$
• 10   endif
  11   return $r$
```

```
$\mathcal{D}4'_{pk}(c)$
   1   if $c \notin \bar{Z}_{pk} \times M_\ell$; return $\perp_1$; endif
   2   $(c_1,c_2) = c$
•  3   $z' \leftarrow \pi'_{pk}(c_1)$
•  4   if $z' \in \mathcal{T}'_{G4}$
•  5       $(x,g) \leftarrow \mathcal{T}'_{G4}(z')$
   6       $m \leftarrow \mathsf{Dec}^{sym}_g(c_2)$
   7       $y \leftarrow H4(x,m)$
   8       if $f_{pk}(x,y) \neq c_1$; return $\perp_2$; endif
   9       return $m$
  10       endif
  11   return $\perp_2$
```

The same standard hashing techniques used in the simulation of G and H can be also used here to maintain $\mathcal{T}'_{G4}$, so the time overhead of step 4 in $\mathcal{D}4'_{pk}$ and step 9 in $G4'$ can be neglected.

Then,

$$\begin{aligned} T[\mathcal{A}_{\mathsf{POW}(f)}] \; &\leq (q_G + q_H + q_D)T[\mathcal{V}] + q_G T[f'] + \\ &+ q_D\Big(T[f] + T[\pi'] + T[\mathsf{Dec}^{sym}]\Big) + T[\mathcal{A}_{\mathsf{IND-CCA}(\mathcal{E})}] \end{aligned}$$