Easy Verifiable Primitives and Practical Public Key Cryptosystems

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 [8] presented a nice generic transformation from weak asymmetric and symmetric schemes into an IND-CCA hybrid encryption scheme in the Random Oracle Model. Two specific candidates for standardization were designed from this transformation: PSEC-2 [14] and EPOC-2 [7], based on El Gamal and Okamoto-Uchiyama 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 prevent such attacks from generic transformation by identifying the properties that an asymmetric scheme must have in order to obtain a secure hybrid scheme. To achieve this, some ambiguities in the proof of the generic transformation [8] which could lead to false claims are described. As a result, the original conversion is modified and the class of asymmetric primitives that can be used is shortened. Secondly, the concept of *Easy Verifiable Primitive* is formalized, showing its connection with Gap problems. Using 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 new conversion's resistance to 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 wishes to fulfil: *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 the attacker uses. The standard security notion for a general purpose cryptosystem is *indistinguishability against adaptive chosen ciphertext attacks*, IND-CCA for short. 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, the concrete security of the scheme; that is, how the security of the scheme is related to the computational assumptions on which it is based. There are other relevant features, such as design simplicity or the length of the messages that can be encrypted.

However, developing a practical provably secure cryptosystem in the sense of IND-CCA is quite a difficult task. In fact, few such schemes are known in the standard model, the schemes designed in the Cramer-Shoup paradigm [5] being the exceptions. In the idealized Random Oracle Model [1], several powerful generic constructions have been designed [8, 15, 12, 3], which provide practical IND-CCA schemes from weak asymmetric and symmetric schemes.

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 [13], known as EPOC-2 [7], has found practical attacks that lead to a total break [10, 6, 16]. The most serious flaw was found in [10], where the secret key was recovered in the IND-CCA game itself. The authors of [10] 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 those ciphertexts. The other attacks mentioned above ([6, 16]), make use of extra information available in the real world, such as the running time of the decryption algorithm. This enables us to distinguish the reasons for rejecting certain ciphertexts, and is used to launch an attack for recovering the secret key.

Our results. We incorporate the comments made by the authors of EPOC about FO conversion in [10]. Then we show that some ambiguities still remain in the security proof, with the outcome that the security result claimed in [8] cannot be in general guaranteed. This entails modifying the conversion slightly and shortening the class of asymmetric primitives that can be used. In the second place, the concept of *Easy Verifiable Primitive* is formalized, and 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, concrete security is improved at the cost of a stronger assumption; that is, a Gap assumption (see [11]). Finally, the resistance of the new conversion to reject timing attacks is addressed. Since the vulnerability of a scheme to these attacks is closely related to the design of the rejection rules in the decryption algorithm, we are careful to take this into consideration when drawing the modification.

Preliminaries. If A is a non-empty set, then $x \leftarrow A$ denotes that x has been uniformly chosen in A. The class of negligible functions on a parameter $\ell \in \mathbb{Z}^+$, denoted as $\mathsf{negl}(\ell)$, is the set of functions $\epsilon : \mathbb{Z}^+ \rightarrow \mathbb{R}^+$ such that, for any polynomial $p \in \mathbb{R}[\ell]$, there exists $M \in \mathbb{R}^+$ such that $\epsilon(\ell) < \frac{M}{p(\ell)}$ for all $\ell \in \mathbb{Z}^+$. Let $\mathsf{poly}(\ell)$ be the class of functions $p : \mathbb{Z}^+ \rightarrow \mathbb{R}^+$ upper bounded in $\mathbb{Z}^+$ by some polynomial in $\mathbb{R}[\ell]$.

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 *polynomial size set* is a set sequence, $X = \{X_\ell\}_{\ell\in\mathbb{Z}^+}$, such that there exists a function $p_X(\ell) \in \mathsf{poly}(\ell)$ and $X_\ell \subseteq \{0,1\}^{p_X(\ell)}$ for all $\ell \in \mathbb{Z}^+$. To simplify the notation, hereafter a *set* will denote a *polynomial size set* and subindexes will be omitted whenever possible. A set X is *samplable* if there exists a probabilistic polynomial time (PPT) algorithm that, on input 1^ℓ, outputs a uniformly distributed random element in X_ℓ. Moreover, X is *recognizable* if there exists a polynomial time (PT) algorithm which, on input 1^ℓ and a string s, outputs 1 if and only if $s \in X_\ell$.

Let PK and SK sets be such that PK_ℓ are all disjoint. Let I be a polynomial time samplable probability distribution over $PK \times SK$. The triple (PK, SK, I) will be called a *keypair generator*. Given a keypair generator, a *set family* X is defined as $\{X_{pk}\}_{pk\in PK}$ and a *map family* $f : X \to Z$ is defined as $\{f_{pk} : X_{pk} \to Z_{pk}\}_{pk\in PK}$.

2 Easy verifiable functions

The following definition, based on [15], is somewhat related to the notion of *probabilistic* one-way (OW) encryption. Let (PK, SK, I) a keypair generator. Let $f : X \times Y \to Z$ be a family of injective maps and $g : Z \to X$ 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 $y \in Y_{pk}$.

Definition 1 *The injective map family, f, is called a* Trapdoor Partial One-Way (TPOW) *function (with respect to the probability distribution I) if*
- *there exists a PT algorithm that on input (pk, x, y) outputs $f_{pk}(x, y)$ for all $pk \in PK$, $x \in X_{pk}$ and $y \in Y_{pk}$.*
- *there exists a PT algorithm that on input (sk, z) outputs $g_{sk}(z)$ for all $sk \in SK$ and for all $z \in Z_{pk}$.*
- *for any PPT algorithm $\mathcal{A}^{\mathrm{POW}}$,*
$\mathsf{Pr}\left[\mathcal{A}^{\mathrm{POW}}(pk, f_{pk}(x, y)) = x \mid (pk, sk) \leftarrow I_\ell; x \leftarrow X_{pk}; y \leftarrow Y_{pk}\right] \in \mathsf{negl}(\ell)$

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 $(pk, sk) = \mathsf{KeyGen}^f(1^\ell)$ 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 $\mathsf{Dec}^f(sk, c) = g_{sk}(c)$.

New kinds of attacks and computational problems have been introduced and various applications found in the context of probabilistic cryptosystems (cf [11, 12]). 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 can be reformulated in terms of trapdoor partial one-way functions.

Definition 2 *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 $\mathcal{O}_{PCA}$ is given. For a query (pk, x, z), where $pk \in PK$, $x \in X_{pk}$ and $z \in Z_{pk}$, $\mathcal{O}_{PCA}$ 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 [15], which captures the situation where there exists an efficient algorithm that *verifies* if a pair (x, z) is correct; that is, the algorithm implements a plaintext checking oracle.

Definition 3 *A map family $f : X \times Y \to Z$ is* easy verifiable *if it is a TPOW family and there exists a (deterministic) PT algorithm $\mathcal{V}$, called* plaintext checking algorithm*, with the same input-output behaviour as the plaintext checking oracle for f.*

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

It is straightforward to modify a trapdoor one-way (TOW) function family $f' : X \to Z'$ to obtain an easy verifiable function family f. To do this, 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 might 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 [11, 12]).

A non-trivial example of an easy verifiable function is the RSA-Paillier trapdoor bijection defined in [2]. A generalization of this function is presented below.

2.1 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

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

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 us 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 5 *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 exists 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

$\Leftarrow$) Trivial. □

Proposition 6 *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, if $z < nr$ and $z \bmod n \in \mathbb{Z}_n^\star$. Then, check if the equation $x^e \equiv z \pmod n$ holds. □

3 Encryption security

Let us briefly recall the security definitions we will consider for symmetric and asymmetric encryption schemes.

3.1 Symmetric encryption

Let K and M be two recognizable sets denoting the keys and messages spaces respectively. Let us 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 a uniformly distributed element in K_ℓ. Enc^{sym} and Dec^{sym} are PT algorithms with inputs 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. Suppose that for each pair $(m, c) \in M_\ell \times M_\ell$ there are at most γ values of $k \in K_\ell$ such that $c = \mathsf{Enc}_k^{sym}(m)$.

Such a cryptosystem has *indistinguishability of encryptions* (IND-SYM), also called Find-Guess security in [8], if any couple of PPT algorithms $\mathcal{A}^{\mathsf{IND-SYM}} = (\mathcal{A}_1, \mathcal{A}_2)$ (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 $\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-SYM}}\right] = |2\mathsf{Pr}\left[b' = b\right] - 1| = |\mathsf{Pr}\left[b' = b\right] - \mathsf{Pr}\left[b' \neq b\right]| \in \mathsf{negl}(\ell)$, for all $(\mathcal{A}_1, \mathcal{A}_2)$. The messages m_0 and m_1 generated by $\mathcal{A}_1$ must be in M_ℓ. Note that this is a very weak security notion, but it suffices to obtain a secure hybrid cryptosystem.

3.2 Asymmetric encryption

Let (PK, SK, I) be a keypair generator, as defined in section 1. Consider an asymmetric encryption scheme $\mathcal{E}^{asym} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, over appropriate sets M, R and C, with the following properties:

- The keys $(pk, sk) = \mathsf{KeyGen}^f(1^\ell)$ are generated by using the sampling algorithm for I.
- Enc is a probabilistic encryption algorithm which, on inputs a public key $pk \in PK$ and $m \in M_{pk}$, runs on a randomness $r \in R_{pk}$ and returns a ciphertext $c \in C_{pk}$.
- Dec is a deterministic decryption algorithm that, on inputs a secret key $sk \in SK$, and c, returns a string m. We require that if $(sk,\ pk) \leftarrow \mathsf{KeyGen}(1^\ell)$, then $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m, r)) = m$ for all $(m, r) \in M_{pk} \times R_{pk}$.

We say $\mathcal{E}^{asym}$ is IND-CCA, if any pair of PPT algorithms $\mathcal{A}^{\mathsf{IND-CCA}} = (\mathcal{A}_1, \mathcal{A}_2)$ have negligible advantage in trying to distinguish the encryptions of two selected messages, with access to a couple of decryption oracles $\mathcal{D}_{sk}$ and $\mathcal{D}_{sk,c^\star}$. When queried with a ciphertext c, the first decryption oracle answers $\mathsf{Dec}(sk, c)$. The only difference between $\mathcal{D}_{sk}$ and $\mathcal{D}_{sk,c^\star}$ is that the second oracle rejects $c^\star$.

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 $b' \leftarrow \mathcal{A}_2^{G, H, \mathcal{D}_{sk,c^\star}}(s, c^\star)$

That is, $\mathcal{E}^{asym}$ is IND-CCA if and only if $\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-CCA}}\right] = |2\mathsf{Pr}\left[b' = b\right] - 1| = |\mathsf{Pr}\left[b' = b\right] - \mathsf{Pr}\left[b' \neq b\right]| \in \mathsf{negl}(\ell)$, for all adversaries $\mathcal{A}^{\mathsf{IND-CCA}}$. The messages m_0 and m_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 implies 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. This information might be very useful for an adversary.

4 Revisiting Fujisaki-Okamoto hybrid scheme

In this section, the transformation introduced in [8] from weak symmetric and asymmetric schemes into an IND-CCA hybrid encryption scheme is revisited.

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, and $\mathcal{E}^{sym} = (\mathsf{KeyGen}^{sym}, \mathsf{Enc}^{sym}, \mathsf{Dec}^{sym})$ be a symmetric encryption scheme. Let $H : X \times M \to Y$ and $G : X \to K$ be random functions. The hybrid scheme $\mathcal{E}^{FO} = (\mathsf{KeyGen}^{FO}, \mathsf{Enc}^{FO}, \mathsf{Dec}^{FO})$ works as follows.

Key generation. The keys (pk, sk) 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 $\mathcal{A}^{\mathsf{IND-CCA}}[T, \epsilon, q_G, q_H, q_D]$ 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 H, G and to the decryption oracle $\mathcal{D}_{sk}$. Then, the result claimed in [8] can be reformulated in the following way:

Theorem 7 *If there exists for some values of ℓ an adversary $\mathcal{A}^{\mathsf{IND-CCA}}[T, \epsilon, q_G, q_H, q_D]$, then there exists an adversary $\mathcal{A}^{\mathrm{POW}}$ that in time T_1 breaks the partial one-wayness of f with success probability ϵ_1 and an adversary $\mathcal{A}^{\mathsf{IND-SYM}}$ that in time T_2 breaks the* IND-SYM *security of $\mathcal{E}^{sym}$ with advantage ϵ_2 such that*

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

$$\text{and} \quad T = \min(T_1, T_2) - O((q_G + q_H)\log(|X||M|))$$

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 [9] this result for the particular case of the Okamoto-Uchiyama scheme [13] (known as EPOC-2) and claimed, without proof, that a tight reduction is obtained for trivial easy verifiable primitives, using our terminology.

4.1 Identifying dangerous ambiguities

However, as pointed out in the introduction, several attacks against EPOC-2 have been found [10, 6, 16]. Despite the refinements introduced in FO conversion after [10], there are still some ambiguities both in the scheme and in the security proof that compromise the validity of the theorem above. For instance, let us consider a TPOW function f, and $X_{pk} \subset \overline{X}_{pk}$ such that $f_{pk}(x, y)$ is computable in polynomial time for any $x \in \overline{X}_{pk}$ and $y \in Y_{pk}$. Then, some badly generated ciphertexts $c = (f_{pk}(x, H(x, m)), \mathsf{Enc}^{sym}_{G(x)}(m))$ for $x \in \overline{X}_{pk} \backslash X_{pk}$ may be accepted. This was the case for the Okamoto-Uchiyama function in the original EPOC-2, where $\overline{X}_{pk} = \mathbb{Z}_{2^{\ell}+1}$ and $X_{pk} = \mathbb{Z}_{2^{\ell}}$, for $2^{\ell} < p < 2^{\ell+1}$. This information was used in [10] to obtain the secret value p.

As Fujisaki and Okamoto proposed later in [9], 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.

Nevertheless, it is possible 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 could only guarantee that breaking security of the cryptosystem is equivalent to solving 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 $\overline{X}_{pk} = \mathbb{Z}_n$. 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 IND-CCA security of the hybrid cryptosystem will be based on the gap between the quadratic residuosity modulo n and factoring n assumptions.

4.2 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 those which admit a correct and unambiguous proof of security for the general transformation. We also take into account the results in [6, 16] 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 [12, 3, 4] 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*, thereby improving the concrete security result presented in the original work.

Let $\mathcal{E}^{sym} = (\mathsf{KeyGen}^{sym}, \mathsf{Enc}^{sym}, \mathsf{Dec}^{sym})$ be a symmetric encryption scheme and $\mathcal{E}^{f} = (\mathsf{KeyGen}^{f}, \mathsf{Enc}^{f}, \mathsf{Dec}^{f})$ be an asymmetric encryption scheme, obtained from a TPOW familiy f. The first change we introduce is that the random functions H, G are defined with inputs in $\{0,1\}^*$ and outputs in their respective

domains, that is, $H : \{0,1\}^* \to Y$ and $G : \{0,1\}^* \to K$. It is unrealistic to restrict the inputs of the random functions, as the authors suggested, since in a practical implementation random functions are replaced by cryptographic hash functions.

Now, X and M must be recognizable sets. Note that this is a restriction only for X, since almost always $M_\ell = \{0,1\}^{p(\ell)}$, for some polynomial p. In contrast, Z is not required 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 change is that now two different rejection symbols are produced in the decryption algorithm Dec. Thus, when a ciphertext is rejected, the adversary will know the reason, obtaining different rejection symbols without mounting a timing attack. Then, if the computing time of each step in the algorithm is independent of the data, the scheme seems to be robust against reject timing attacks.

$\mathsf{Dec}(sk, c)$
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

It is necessary to 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 from detecting an extra rejection reason.

Next the security results are stated. The first theorem is for the special case where f is an easy verifiable function, while the corollary works for general TPOW functions.

Theorem 8 *If there exists for some values of ℓ an adversary $\mathcal{A}^{\mathsf{IND-CCA}}[T, \epsilon, q_G, q_H, q_D]$ against the* IND-CCA *security of the proposed cryptosystem for an easy verifiable function family f, then there exists an adversary $\mathcal{A}^{\mathrm{POW}}$ that in time T_1 breaks the partial one-wayness of f with success probability ϵ_1, and an adversary $\mathcal{A}^{\mathsf{IND-SYM}}$ that in time T breaks the IND-SYM security of $\mathcal{E}^{sym}$ with advantage ϵ_2 such that*

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

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

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

Proof: The proof is delayed for the appendix. □

Notice that now the probabilities are tightly related. In the general case, a plaintext checking algorithm might not exist. Using the access to a plaintext checking oracle instead, the following result is straightforward.

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

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

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

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

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

4.3 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 2.1, the simulation in the security proof can be improved by introducing few technical modifications.

In both cases, there exist two very efficiently computable functions $\tilde{f}_{pk} : X_{pk} \to \tilde{Z}_{pk}$ and $\tilde{\pi}_{pk} : Z_{pk} \to \tilde{Z}_{pk}$ such that $\mathcal{V}(pk, x, z) = 1$ if and only if $\tilde{f}_{pk}(x) = \tilde{\pi}_{pk}(z)$. Then, it is shown in the appendix that

$$\begin{aligned} T[\mathcal{A}^{\mathrm{POW}}] \quad &\leq (q_G + q_H + q_D)T[\mathcal{V}] + q_G T[\tilde{f}] + \\ &\quad + q_D\Big(T[f] + T[\tilde{\pi}] + T[\mathsf{Dec}^{sym}]\Big) + T[\mathcal{A}^{\mathsf{IND-CCA}}], \end{aligned}$$

providing a *very tight* security reduction.

If the trivial constructions are considered, $f_{pk}(x, y) = (\tilde{f}_{pk}(x), y)$ and $\tilde{\pi}_{pk}(\tilde{z}, y) = \tilde{z}$ so $T[\tilde{\pi}]$ can be neglected. Moreover, $T[f] \approx T[\tilde{f}] \approx T[\mathcal{V}]$ so

$$T[\mathcal{A}^{\mathrm{POW}}] \leq (2q_G + q_H + 2q_D)T[\tilde{f}] + q_D T[\mathsf{Dec}^{sym}] + T[\mathcal{A}^{\mathsf{IND-CCA}}]$$

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

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

Acknowledgements

The authors thank the anonymous referees for their valuable comments on this paper.

References

1. M. Bellare and P. Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. *ACM CCS 93, ACM Press* (1993)
2. D. Catalano, R. Gennaro, N. Howgrave-Graham and P. Q. Nguyen. Paillier's Cryptosystem Revisited. *ACM CCS '2001 ACM Press* (2001).
3. 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).
4. 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).
5. 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).
6. A. W. Dent. An implementation attack against the EPOC-2 public-key cryptosystem. *Electronics Letters* VOL. 38 NO. 9 412–413 (2002).
7. EPOC, Efficient Probabilistic Public-Key Encryption. `http://info.isl.ntt.co.jp/epoc/`
8. E. Fujisaki and T. Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. *CRYPTO '99, LNCS* **1666** 537–554 (1999)
9. 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).
10. 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).
11. 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).
12. T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. *CT-RSA' 01, LNCS* **2020** 159–175 (2001).
13. T. Okamoto and S. Uchiyama. A New Public-Key Cryptosystem as Secure as Factoring. *EUROCRYPT-98, LNCS* **1403** 308–318 (1998)
14. PSEC, Provably Secure Encryption Scheme. `http://info.isl.ntt.co.jp/psec/`
15. D. Pointcheval. Chosen-Ciphertext Security for any One-Way Cryptosystem. *Proc. PKC '2000 LNCS* **1751** 129–146 (2000).
16. K. Sakurai and T. Takagi. A Reject Timing Attack on an IND-CCA2 Public-Key Cryptosystem. *ICISC '02, LNCS* **2587** 359–373 (2002).

A Proof of theorem 8

Let $\mathcal{A}^{\mathsf{IND-CCA}} = (\mathcal{A}_1, \mathcal{A}_2)$ 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 4.2.

In order to prove the theorem, some different games will be considered. In all games, the adversary $\mathcal{A}^{\mathsf{IND-CCA}}$ uses the same coins, but the events defined as functions of the view of $\mathcal{A}^{\mathsf{IND-CCA}}$ could occur with different probabilities in each game. Starting from the IND-CCA game, we will decribe 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. The following trivial lemma will be very useful in this proof.

Lemma 10 *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 3.2, but they do not modify any probability.

Game0()
1 $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell)$; $G, H \leftarrow \mathcal{R}$
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 $y^\star \leftarrow H(x^\star, m_b)$; $c^\star \leftarrow \left(f_{pk}(x^\star, y^\star), \mathsf{Enc}^{sym}_{G(x^\star)}(m_b)\right)$
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 4.2.

Let S_1 be the event where, during the game, either $x^\star$ is queried (by $\mathcal{A}^{\mathsf{IND-CCA}}$) to G or $(x^\star, m)$ is queried to H, for some m. Let us also define $\mathsf{S}_{01} = \neg\mathsf{S}_1 \wedge (b' = b)$ and $\mathsf{S}_{00} = \neg\mathsf{S}_1 \wedge (b' \neq b)$. After some easy computations, we obtain

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

Let $\mathcal{T}_G$ be a table in which all queries made by $\mathcal{A}^{\mathsf{IND-CCA}}$ to the oracle G are stored along with the corresponding answers. Define $\mathcal{T}_H$ and $\mathcal{T}_\mathcal{D}$ in a similar way for the oracle calls to H and $\mathcal{D}_{sk}$ respectively. Notice that the contents of these tables change during the game.

Game1. In this game, the queries made by $\mathcal{A}^{\mathsf{IND-CCA}}$ to the two random oracles are intercepted in order to abort the execution of the game immediately if S_1 occurs.

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

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

Game2. In this game, the decryption oracle is modified in such a way that new queries to the random oracle G are disallowed. 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 $\perp_2$, even when some of them may be valid ciphertexts.

Let F_2 be the event where, in some query to the decryption oracle, the ciphertext is accepted in Game1 but rejected by the decryption oracle in Game2. Before F_2 occurs, both games are indentical. Then, by lemma 10,

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

From these inequalities, it can be easily shown that

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

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

Lemma 11

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

(The proof of this lemma can be found in the full version of the paper.)

Game2′. In this game, oracles G and H are simulated by using tables. The answers to new queries to G or H are perfectly simulated by uniformly distributed independent random variables. The generation of the ciphertext, which is also different, is done as follows:

$$g \leftarrow K_\ell;\ y^\star \leftarrow Y_\ell;\ c^\star \leftarrow \left(f_{pk}(x^\star, y^\star), \mathsf{Enc}_g^{sym}(m_b)\right)$$

which is equivalent to redefining some values of the random functions, 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 $\mathcal{A}^{\mathsf{IND-CCA}}$, since in Game2 neither $x^\star$ is queried to G nor $(x^\star, m)$ is queried to H, for any m.

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 $\mathcal{A}^{\mathsf{IND-SYM}} = (\mathcal{A}_1^{sym}, \mathcal{A}_2^{sym})$ who tries to break the IND security of $\mathcal{E}^{sym}$ (see 3.1).

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 $\mathcal{T}_{G3} \leftarrow \mathsf{empty}$; $\mathcal{T}_{H3} \leftarrow \mathsf{empty}$
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, σ)

$\mathcal{A}_2^{sym}(\sigma, \kappa^\star)$
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 $\quad \beta' \leftarrow 0$
7 else
8 $\quad \beta' \leftarrow 1$
9 endif

In addition, if S_1 occurs, then $\beta' \leftarrow \{0,1\}$ and $\beta'' \leftarrow 1$ before exiting the game in the simulators of G and H.

Now, two different advantages can be taken into account: $\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-SYM}}\right] = |2\mathsf{Pr}_3[\beta' = \beta] - 1|$ and $\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-SYM}}\right]' = |2\mathsf{Pr}_3[\beta'' = \beta] - 1|$. It can be proved that

$$\begin{aligned}\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-CCA}}\right] \quad &\leq \mathsf{Pr}_3[\mathsf{S}_1 \mid \beta = 1] + 2\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-SYM}}\right] + \\ &\quad + \mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-SYM}}\right]' + 2\mathsf{Pr}[F_2]\end{aligned}$$

Game4. Game3 (with $\beta = 1$) can be modified to obtain an implementation of an adversary, $\mathcal{A}^{\mathrm{POW}}$, which tries 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 is avoided by using the deterministic plaintext checking algorithm $\mathcal{V}$.

Game4()
1 $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\ell)$
2 $x^\star \leftarrow X_{pk};\ y^\star \leftarrow Y_{pk};\ z \leftarrow f_{pk}(x^\star, y^\star)$
3 $\mathcal{A}^{\mathrm{POW}}(pk, z^\star)$

$\mathcal{A}^{\mathrm{POW}}(pk, z)$
1 $b \leftarrow \{0,1\}$
2 $m \leftarrow M_\ell;\ g \leftarrow K_\ell;\ c^\star \leftarrow (z, \mathsf{Enc}_g^{sym}(m))$
3 $\mathcal{T}_G \leftarrow$ empty ; $\mathcal{T}_H \leftarrow$ empty
4 $(m_0, m_1, s) \leftarrow \mathcal{A}_1^{G,H,\mathcal{D}4_{pk}}(pk)$
5 $b' \leftarrow \mathcal{A}_2^{G,H,\mathcal{D}4_{pk,c^\star}}(s, c^\star)$
6 $x' \leftarrow X_{pk}$

$\mathcal{D}4_{pk}(c)$
1 if $c \notin \bar{Z}_{pk} \times M_\ell$; return $\perp_1$; endif
2 $(c_1, c_2) = c$
3 foreach $x \in \mathcal{T}_{G4}$
4 $\quad$ if $x \in X_{pk}$ and $\mathcal{V}(pk, x, c_1) = 1$
5 $\quad\quad m \leftarrow \mathsf{Dec}^{sym}_{\mathcal{T}_{G4}(x)}(c_2)$
6 $\quad\quad y \leftarrow H4(x, m)$
7 $\quad\quad$ if $f_{pk}(x, y) \neq c_1$; return $\perp_2$; endif
8 $\quad\quad$ return m
9 $\quad$ endif
10 endforeach
11 return $\perp_2$

The use of $x^\star$ in the random oracle simulators is also avoided. To do this, S_1 is detected by checking if $x \in X_{pk} \wedge \mathcal{V}(pk, x, z) = 1$ for each query x or (x, m). If S_1 occurs then $x' \leftarrow x$.

These changes do not modify any probability. Moreover, the views of $\mathcal{A}^{\mathsf{IND-CCA}}$ in games 3 (with $\beta = 1$) and 4 are identically distributed. Then,

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

and, from the above results,

$$\begin{aligned}\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-CCA}}\right] \quad &\leq \mathsf{Succ}\left[\mathcal{A}^{\mathrm{POW}}\right] + 2\mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-SYM}}\right] + \\ &\quad + \mathsf{Adv}\left[\mathcal{A}^{\mathsf{IND-SYM}}\right]' + \frac{2q_D q_H \gamma}{|K| - q_D q_H \gamma} + \frac{2q_D}{|Y| - q_D}\end{aligned}$$

In terms of time complexity of the algorithms, it is supposed that the time needed to check if $c \in \bar{Z}_{pk} \times M_\ell$ and $x \in X_{pk}$ is negligible.

Neglecting lower order terms, the running time of $\mathcal{A}^{\mathrm{POW}}$ in Game4 is bounded by

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

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

A.1 Particular cases

Both in the case of the trivial construction of easy verifiable functions and in the non-trivial family in subsection 2.1, the algorithm $\mathcal{D}4_{pk}$ can be improved, without modifying the behavior of the game, to avoid exhaustive search in $\mathcal{T}_{G4}$. If $(\tilde{f}(x), x, G(x))$ is stored in another table $\tilde{\mathcal{T}}_{G4}$ for each query $x \in X_{pk}$ to G, the decryption oracle can be simulated as follows:

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

Now,

$$\begin{aligned}T[\mathcal{A}^{\mathrm{POW}}] \quad &\leq (q_G + q_H + q_D)T[\mathcal{V}] + q_G T[\tilde{f}] + \\ &\quad + q_D\Big(T[f] + T[\tilde{\pi}] + T[\mathsf{Dec}^{sym}]\Big) + T[\mathcal{A}^{\mathsf{IND-CCA}}]\end{aligned}$$