Optimal Chosen-Ciphertext Secure Encryption of Arbitrary-Length Messages

Jean-Sébastien Coron[1], Helena Handschuh[1], Marc Joye[2], Pascal Paillier[1], David Pointcheval[3], and Christophe Tymen[1,3]

[1] Gemplus Card International
34 rue Guynemer, 92447 Issy-les-Moulineaux, France
{jean-sebastien.coron, helena.handschuh, pascal.paillier, christophe.tymen}@gemplus.com

[2] Gemplus Card International
Parc d'Activités de Gémenos, B.P. 100, 13881 Gémenos Cedex, France
marc.joye@gemplus.com – http://www.geocities.com/MarcJoye/

[3] École Normale Supérieure, Computer Science Department
45 rue d'Ulm, 75230 Paris Cedex 05, France
david.pointcheval@ens.fr – http://www.di.ens.fr/~pointche/

Abstract. This paper considers arbitrary-length chosen-ciphertext secure asymmetric encryption, thus addressing what is actually needed for a practical usage of strong public-key cryptography in the real world. We put forward two generic constructions, GEM-1 and GEM-2, which apply to explicit *fixed-length* weakly secure primitives and provide a strongly secure (IND-CCA2) public-key encryption scheme for messages of unfixed length (typically computer files). Our techniques optimally combine a single call to any one-way trapdoor function with repeated encryptions through some weak block-cipher (a simple XOR is fine) and hash functions of fixed-length input so that a minimal number of calls to these functions is needed. Our encryption/decryption throughputs are comparable to the ones of standard methods (asymmetric encryption of a session key + symmetric encryption with multiple modes). In our case, however, we *formally* prove that our designs are secure in the strongest sense and provide complete security reductions holding in the random oracle model.

1 Introduction

A real-life usage of public-key encryption requires three distinct ideal properties. *Security* is a major concern: a cryptosystem should be secure against any attack of any kind, should the attack be realistic in the context of use or only theoretical. *Performance* has to be risen to upmost levels to guarantee high speed encryption *and* decryption rates in communication protocols and real time applications. At last, *design simplicity* is desirable to save time and efforts in software or hardware developments, increase modularity and reusability, and facilitate public understanding and scrutiny.

Designing an encryption scheme which meets these criteria is quite a challenging work, but methodologies and tools exist, at least for the first property. Our knowledge of security features inherent to cryptographic objects and of the relations connecting them has intensively evolved lately, driving us to a growing range of powerful generic constructions, both simple and provably secure [8, 9, 14, 13]. Among these constructions, Okamoto and Pointcheval's REACT [13] is certainly the one that offers most flexibility: unlike Bellare and Rogaway's long-lived OAEP [5], REACT applies to any trapdoor function, *i.e.* any asymmetric encryption scheme presenting such a weak level of security as being OW-PCA (see further), to provide a cryptosystem of strongest level IND-CCA2 in the random oracle model.

1.1 Our results

This paper considers arbitrary-length chosen-ciphertext secure (IND-CCA2) asymmetric encryption schemes, thus addressing what is actually needed for a practical usage of strong public-key cryptography in the real world. We propose two generic constructions which apply to fixed-length weakly secure primitives and provide a strongly secure public-key cryptosystem for messages of unfixed length, such as computers files or communication streams. In our schemes, the encryption and decryption processes may start and progress without even knowing the overall input blocklength; they also may stop at any time. Besides, our designs are one-pass only, meaning that each message block will be treated exactly once.

Our techniques combine a *single* call to any one-way trapdoor function with repeated encryptions through some weak block-cipher (a simple XOR will do) and hash functions of fixed-length input. Contrarily to previous generic conversions, each message block will require only *one* call to a hash function so that the overall execution cost for an n-block plaintext is exactly 1 call to the one-way trapdoor encryption, followed by n calls to the block-cipher and $n + 1$ (or $n + 2$) calls to a hash function[1]. Besides, the storage of the whole plaintext file in memory is completely unnecessary and encryption/decryption procedures use a memory buffer of only ~ 3 blocks, thus allowing on-the-fly treatments of communication streams. We believe that our schemes are the first that combine these practical properties simultaneously while keeping total genericity.

The first construction applies to any OW-PCA probabilistic trapdoor function and incorporates two extra fields of fixed length in the ciphertext, one at each end. The second construction we give only works for deterministic OW trapdoor functions but adds only one extra field at the end of the ciphertext. Our performances are similar to the ones of standard methods, which usually encrypt some random session key under an asymmetric scheme and then feed that key into some block-cipher running under an appropriate multiple mode. In our case, however, we can formally prove that our designs are secure in the strongest sense IND-CCA2. Indeed, we provide complete security reductions holding in the random oracle model.

[1] an extra hash call is needed for authenticity.

1.2 Outline of the paper

The paper is organized as follows. Section 2 briefly recalls security notions for encryption schemes in both symmetric and asymmetric settings. We also review Okamoto and Pointcheval's plaintext checking attacks in connection with computational gap problems [12]. Then, Sections 3.1 and 3.2 introduce our new generic conversions, GEM-1 and GEM-2, whose reduction proofs are given in Appendices B and C. Furthermore, typical examples of practical usage of these systems are given in Section 4. We conclude by giving some possible extensions of our work in Section 5.

2 Security notions for encryption schemes

2.1 Asymmetric encryption

We now introduce a few standard notations. An asymmetric encryption scheme is a triple of algorithms $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ where

- $\mathcal{K}$ is a probabilistic key generation algorithm which returns random pairs of secret and public keys (sk, pk) depending on the security parameter κ,
- $\mathcal{E}$ is a probabilistic encryption algorithm which takes on input a public key pk and a plaintext $m \in \mathcal{M}$, runs on a random tape $u \in \mathcal{U}$ and returns a ciphertext c,
- $\mathcal{D}$ is a deterministic decryption algorithm which takes on input a secret key sk, a ciphertext c and returns the corresponding plaintext m or the symbol $\perp$. We require that if $(sk, pk) \leftarrow \mathcal{K}$, then $\mathcal{D}_{sk}(\mathcal{E}_{pk}(m, u)) = m$ for all $(m, u) \in \mathcal{M} \times \mathcal{U}$.

Adversarial goals.

ONE-WAYNESS. The first secrecy notion required from an encryption scheme is its *one-wayness*, meaning that one should not be able to recover a plaintext given its encryption. More formally, the scheme is said to be (τ, ε)-OW if for any adversary $\mathcal{A}$ with running time bounded by τ, the probability that $\mathcal{A}$ inverts $\mathcal{E}$ is less than ε:

$$\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A}) = \Pr_{\substack{m \xleftarrow{R} \mathcal{M} \\ u \xleftarrow{R} \mathcal{U}}} [(sk, pk) \leftarrow \mathcal{K}(1^\kappa) : \mathcal{A}(\mathcal{E}_{pk}(m, u)) = m] < \varepsilon ,$$

where the probability is taken over the random choices of the adversary.

SEMANTIC SECURITY. Formalizing another security criterion that an encryption scheme should verify beyond one-wayness, Goldwasser and Micali [10] introduced the notion of *semantic security*. Also called *indistinguishability of encryptions* (or IND for short), this property captures the idea that an adversary should not be able to learn any information whatsoever about a plaintext, its length excepted, given its encryption. More formally, an asymmetric encryption scheme is said to be (τ, ε)-IND if for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with running time upper-bounded by τ,

$$\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) = 2 \times \Pr_{\substack{b \xleftarrow{R} \{0,1\} \\ u \xleftarrow{R} \mathcal{U}}} \left[\begin{array}{l} (sk, pk) \leftarrow \mathcal{K}(1^\kappa), (m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(pk) \\ c \leftarrow \mathcal{E}_{pk}(m_b, u) : \mathcal{A}_2(c, \sigma) = b \end{array} \right] - 1 < \varepsilon ,$$

where the probability is taken over the random choices of $\mathcal{A}$. The two plaintexts m_0 and m_1 chosen by the adversary in $\mathcal{M}$ have to be of identical length.

Non-malleability. The property of *non-malleability* (NM), independently proposed by Dolev, Dwork and Naor [6], supposes that, given the encryption of a plaintext m, the attacker cannot produce the encryption of a related plaintext m'. Here, rather than learning some information about m, the adversary will try to output the encryption of m'. These two properties are related in the sense that non-malleability implies semantic security for any adversarial model, as pointed out in [6] and [3].

Adversarial models. On the other hand, there exist several types of adversaries, or attack models. In a chosen-plaintext attack (CPA), the adversary has access to an encryption oracle, hence to the encryption of any plaintext she wants. Clearly, in the public-key setting, this scenario cannot be avoided. Naor and Yung [11] considered non-adaptive chosen-ciphertext attacks ($\mathsf{CCA1}$) (also known as lunchtime or midnight attacks), wherein the adversary gets, in addition, access to a decryption oracle before being given the challenge ciphertext. Finally, Rackoff and Simon [15] defined adaptive chosen-ciphertext attacks ($\mathsf{CCA2}$) as a scenario in which the adversary queries the decryption oracle before and *after* being challenged; her only restriction here is that she may not feed the oracle with the challenge ciphertext itself. This is the strongest known attack scenario.

Various security levels are then defined by pairing each goal (OW, IND or NM) with an attack model (CPA, $\mathsf{CCA1}$ or $\mathsf{CCA2}$), these two characteristics being considered separately. Interestingly, it has been shown that $\mathsf{IND\text{-}CCA2}$ and $\mathsf{NM\text{-}CCA2}$ were strictly equivalent notions [3]. This level is now considered as standard and referred to as $\mathsf{IND\text{-}CCA2}$ security or chosen-ciphertext security. The security of a cryptosystem is thus measured as the ability to resist an adversarial goal in a given adversarial model. Whenever possible, the scheme is proven $\mathsf{IND\text{-}CCA2}$ secure by exhibiting a polynomial reduction: if some adversary can break the $\mathsf{IND\text{-}CCA2}$ security of the system, then the same adversary can be invoked (polynomially many times) to solve some related hard problem.

2.2 Symmetric encryption schemes

A symmetric encryption scheme with key bit-length $\mathtt{k}$ and message bit-length $\mathtt{m}$ is a pair of algorithms $(\mathtt{E}, \mathtt{D})$ where

– $\mathtt{E}$ is a deterministic encryption algorithm which takes a key $k \in \{0,1\}^{\mathsf{k}}$ and a plaintext $m \in \{0,1\}^{\mathsf{m}}$ and returns a ciphertext $c \in \{0,1\}^{\mathsf{m}}$,
– $\mathtt{D}$ is a deterministic decryption algorithm which takes a key $k \in \{0,1\}^{\mathsf{k}}$ and a ciphertext $c \in \{0,1\}^{\mathsf{m}}$ and returns a plaintext $m \in \{0,1\}^{\mathsf{m}}$. We require that $\mathtt{D}_k(\mathtt{E}_k(m)) = m$ for all $m \in \{0,1\}^{\mathsf{m}}$ and $k \in \{0,1\}^{\mathsf{k}}$.

In this setting, again, various security notions are defined; most are adaptations from the asymmetric notions. In this work, however, we only need to define indistinguishability. A symmetric encryption scheme is said (τ, ε)-IND if for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with running time bounded by τ,

$$\mathsf{Adv}^{\mathsf{ind}}(\mathcal{A}) = 2 \times \Pr_{\substack{k \xleftarrow{R} \{0,1\}^{\mathsf{k}} \\ b \xleftarrow{R} \{0,1\}}} [(m_0, m_1, \sigma) \leftarrow \mathcal{A}_1(\mathsf{k}), c \leftarrow \mathtt{E}_k(m_b) : \mathcal{A}_2(c, \sigma) = b] - 1 < \varepsilon ,$$

where the probability is also taken over the random choices of $\mathcal{A}$. Both plaintexts m_0 and m_1 are chosen by the adversary in $\{0,1\}^{\mathsf{k}}$. Although other attack scenarios may be considered, passive attacks are enough for our purposes. Note that this notion is a very weak requirement. Note also that the one-time pad encryption is perfectly indistinguishable, *i.e.* , it is $(\tau, 0)$-IND for any τ.

2.3 Plaintext-checking security

Okamoto and Pointcheval recently introduced an intermediate adversarial model called plaintext checking attacks [13]. In this model, the adversary has access to a *plaintext-checking* oracle $\mathcal{O}^{\text{PCA}}$ which detects plaintext-ciphertext correspondences: the oracle takes as input a pair (m, c) and tells whether c encrypts m or not. Clearly, this oracle remains weaker than a decryption oracle because it is generally easier to check the solution of a problem (scheme inversion here) than to compute it. Obviously in the case of a deterministic encryption scheme, PCA and CPA are strictly equivalent attack scenarios. More specifically, any trapdoor permutation is OW-PCA if and only if it is OW (e.g., RSA).

From a complexity viewpoint, breaking a scheme's OW-PCA-security exactly consists in breaking its OW-security (*i.e.* its one-wayness) with the help of an oracle solving a weaker problem. That kind of problems, *i.e.* solving P_1 with access to $\mathcal{O}^{\mathrm{P}_2}$ and $\mathrm{P}_2 \Leftarrow \mathrm{P}_1$, are called *gap problems* [12] and define some notion of complexity distance between problems in a hierarchy.

A typical example is ElGamal encryption, for which breaking OW is equivalent to CDH and having access to $\mathcal{O}^{\text{PCA}}$ allows to solve DDH trivially (and conversely). OW-PCA-security is in this case equivalent to the gap problem separating CDH from DDH, which is called *Gap Diffie-Hellman Problem* and noted GDH (see [12] for insights).

2.4 Generic conversions

In [5], Bellare and Rogaway proposed OAEP, a specific hash-based treatment applicable to any *partial-domain* [16, 7] one-way trapdoor permutation to provide an IND-CCA2 secure encryption scheme in the random oracle model [4]. Later, Fujisaki and Okamoto [8] presented a way to transform, still in the random oracle model, any IND-PCA trapdoor function into an IND-CCA2 encryption scheme. They improved their results in [9] where they gave a generic method to convert a *one-way* trapdoor function into an IND-CCA2 secure encryption scheme in the random oracle model[2]. A similar result was independently discovered by Pointcheval [14]. More recently, Okamoto and Pointcheval [13] proposed a more efficient generic conversion, called REACT. Contrarily to [8, 9, 14], a complete re-encryption is unnecessary in the decryption process of REACT to ensure IND-CCA2 security, thus yielding a low running time overhead. Besides, REACT applies to any trapdoor function *i.e.* any asymmetric encryption scheme presenting such a weak level of security as being OW-PCA. Until now, however, no generic conversion has been explicitly defined[3] to encrypt messages of variable length based on fixed-length functions. The next section describes our arbitrary-length generic conversions.

3 Arbitrary-length IND-CCA2 encryption

The most popular and usual way of ensuring confidentiality of unfixed-length messages consists in public-key encrypting a random session key and then encrypting the message under that session key by the means of a block-cipher used within a suitable encryption mode. This approach has never been shown secure; in particular, the use of an IND-CCA2 asymmetric scheme to encrypt the session key is obviously insufficient to ensure any security whatsoever about the whole construction.

In comparison, our conversions are based on the same primitives, *i.e.* some asymmetric scheme $\mathcal{E}_{pk}$ and some symmetric scheme $\mathtt{E}_k$. But we additionally use hash functions to make the session key evolve permanently as the encryption progresses. Our important result here is that the two cryptosystems we propose are IND-CCA2-secure provided that $\mathcal{E}_{pk}$ is OW-PCA or OW and $\mathtt{E}_k$ is indistinguishable. Independently, they provide different security/performance tradeoffs that we analyze in section 4.

3.1 Relying on a OW-PCA trapdoor function: GEM-1

Our first construction $\mathbb{E}^1_{pk}$ applies to any OW-PCA probabilistic trapdoor function $\mathcal{E}_{pk}$ and incorporates two extra fields of fixed length in the ciphertext, one at each end. To make the security proof easier, we will assume that the message blocklength is upper-bounded by some very large number n_{max} which value is discussed in section 4. The encryption and decryption procedures are as depicted below.

[2] the conversion cost is however quite heavy as a complete re-encryption is needed during decryption.

[3] note that [13] considers the case of variable-length encryption without providing any explicit construction for fixed-length functions.

Encryption

Input plaintext $(m_1, \ldots, m_n)$, $1 \leq n \leq n_{max}$, random $\rho = w \| u$.
Output ciphertext $(t_1, c_1, \cdots, c_n, t_2)$ given by

$$\mathbb{E}^1_{pk}(m, \rho) = (\underbrace{\mathcal{E}_{pk}(w, u)}_{t_1}, \underbrace{\mathtt{E}_{k_1}(m_1)}_{c_1}, \underbrace{\mathtt{E}_{k_2}(m_2)}_{c_2}, \cdots, \underbrace{\mathtt{E}_{k_n}(m_n)}_{c_n}, \underbrace{\mathrm{F}(k_n, m_n, w)}_{t_2})$$

where $k_1 = \mathrm{H}_1(w, t_1)$, $k_2 = \mathrm{H}_2(k_1, m_1, w)$, $\ldots$, $k_n = \mathrm{H}_n(k_{n-1}, m_{n-1}, w)$.

Decryption

Input ciphertext $(t_1, c_1, \cdots, c_n, t_2)$ with $1 \leq n \leq n_{max}$.
Output plaintext $(\hat{m}_1, \cdots, \hat{m}_n)$ or $\perp$ according to

$$\mathbb{D}^1_{sk}(t_1, c_1, \cdots, c_n, t_2) = \begin{cases} \hat{m}_1 = \mathtt{D}_{\hat{k}_1}(c_1), \ldots, \hat{m}_n = \mathtt{D}_{\hat{k}_n}(c_n) & \text{if } t_2 = \mathrm{F}(\hat{k}_n, \hat{m}_n, \hat{w}) \\ \perp & \text{otherwise} \end{cases}$$

where $\hat{w} = \mathcal{D}_{sk}(t_1)$, $\hat{k}_1 = \mathrm{H}_1(\hat{w}, t_1)$ and $\hat{k}_i = \mathrm{H}_i(\hat{k}_{i-1}, \hat{m}_{i-1}, \hat{w})$ for $i = 2, n$.

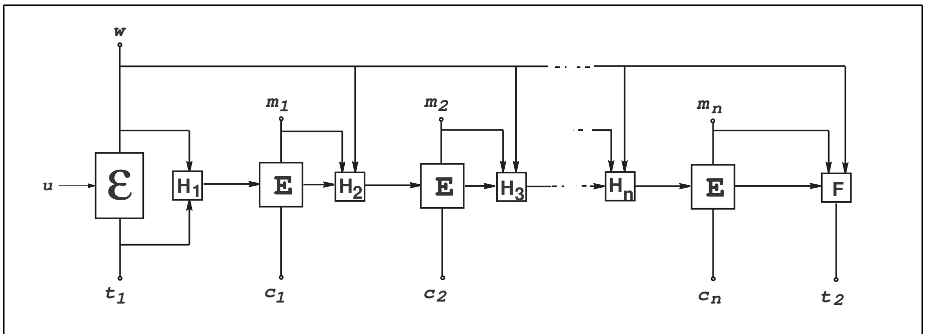

Fig. 1. Synopsis of GEM-1.

We claim that for any OW-PCA asymmetric encryption $\mathcal{E}_{pk}$ and any IND-secure symmetric encryption scheme $\mathtt{E}_k$, our converted scheme $\mathbb{E}^1_{pk}[\mathcal{E}_{pk}, \mathtt{E}_k]$ is IND-CCA2 in the random oracle model. To be more precise:

Theorem 1. *Suppose there exists an adversary $\mathcal{A}$ which distinguishes $\mathbb{E}^1_{pk}[\mathcal{E}_{pk}, \mathtt{E}_k]$ within a time bound τ with advantage ε in less than q_{F}, $q_{\mathrm{H}} = \sum_{i \in \langle 1, n_{max} \rangle} q_{\mathrm{H}_i}$, $q_{\mathbb{D}^1_{sk}}$ oracle calls. Suppose also that $\mathtt{E}_k$ is (τ, ν)-indistinguishable. Then there exists an algorithm $\mathcal{B}$ which inverts $\mathcal{E}_{pk}$ with probability ε' greater than*

$$\varepsilon' \geq \frac{\varepsilon}{2} - q_{\mathbb{D}^1_{sk}} \left(\frac{1}{\sharp\, t_2} + \frac{3}{\sharp\, k} \right) - n_{max} \left(\frac{\nu}{2} + \frac{q_{\mathbb{D}^1_{sk}}}{\sharp\, k} \right) ,$$

with a total number of calls to $\mathcal{O}^{\mathrm{PCA}}$ upper-bounded by $q_{\mathcal{O}^{\mathrm{PCA}}} \leq q_{\mathrm{F}} + q_{\mathrm{H}}$ and in time

$$\tau_{\mathcal{B}} = \tau + (q_{\mathbb{D}^1_{sk}} + 1)(q_{\mathrm{F}} + q_{\mathrm{H}}) \cdot (\tau_{\mathrm{PCA}} + O(1)) .$$

Here, $\sharp\, a$ denotes the number of all possible values of a (hence $\sharp\, k = 2^{\mathsf{k}}$).

We refer the reader to the (extensive) reduction proof given in appendix B.

3.2 Relying on a OW trapdoor function: GEM-2

Our second construction $\mathbb{E}^2_{pk}$ only works with a deterministic OW trapdoor function $\mathcal{E}_{pk}$ (such as RSA) but adds only one extra field at the end of the ciphertext. Here again, we will assume that the message blocklength is upper-bounded by some large number n_{max}. The encryption and decryption procedures follow.

Encryption

Input plaintext $(m_1, \ldots, m_n)$, $1 \le n \le n_{max}$, random r.
Output ciphertext $(c_1, \cdots, c_n, t)$ given by

$$\mathbb{E}^2_{pk}(m, r) = (\underbrace{\mathtt{E}_{k_1}(m_1)}_{c_1}, \underbrace{\mathtt{E}_{k_2}(m_2)}_{c_2}, \cdots, \underbrace{\mathtt{E}_{k_n}(m_n)}_{c_n}, \underbrace{\mathcal{E}_{pk}(s \| v)}_{t})$$

$$\text{where} \begin{cases} k_1 = \mathrm{G}_1(r), k_i = \mathrm{G}_i(k_{i-1}, m_{i-1}, r) \text{ for } i = 2, \ldots n, \\ s = \mathrm{F}(k_n, m_n, r), \text{ and } v = r \oplus \mathrm{H}(s). \end{cases}$$

Decryption

Input ciphertext $(c_1, \cdots, c_n, t)$ with $1 \le n \le n_{max}$.
Output plaintext $(\hat{m}_1, \cdots, \hat{m}_n)$ or $\perp$ according to

$$\mathbb{D}^2_{sk}(c_1, \cdots, c_n, t) = \begin{cases} \hat{m}_1 = \mathtt{D}_{\hat{k}_1}(c_1), \ldots, \hat{m}_n = \mathtt{D}_{\hat{k}_n}(c_n) & \text{if } \hat{s} = \mathrm{F}(\hat{k}_n, \hat{m}_n, \hat{r}), \\ \perp & \text{otherwise.} \end{cases}$$

$$\text{where} \begin{cases} \hat{s} \| \hat{v} = \mathcal{D}_{sk}(t), \hat{r} = \hat{v} \oplus \mathrm{H}(\hat{s}), \\ \hat{k}_1 = \mathrm{G}_1(\hat{r}), \text{and } \hat{k}_i = \mathrm{G}_i(\hat{k}_{i-1}, \hat{m}_{i-1}, \hat{r}) \text{ for } i = 2, \ldots n. \end{cases}$$

We claim that for any OW asymmetric encryption $\mathcal{E}_{pk}$ and any IND-secure symmetric encryption scheme $\mathtt{E}_k$, the converted scheme $\mathbb{E}^2_{pk}[\mathcal{E}_{pk}, \mathtt{E}_k]$ is IND-CCA2 in the random oracle model. To be more precise:

Theorem 2. *Suppose there exists an adversary $\mathcal{A}$ which distinguishes $\mathbb{E}^2_{pk}[\mathcal{E}_{pk}, \mathtt{E}_k]$ within a time bound τ with advantage ε in less than q_F, q_H, $q_\mathrm{G} = \sum_{i \in \langle 1, n_{max} \rangle} q_{\mathrm{G}_i}$, $q_{\mathbb{D}^2_{sk}}$ oracle calls. Suppose also that $\mathtt{E}_k$ is (τ, ν)-indistinguishable. Then there exists an algorithm $\mathcal{B}$ which inverts $\mathcal{E}_{pk}$ with probability ε' greater than*

$$\varepsilon' \ge \frac{\varepsilon}{2} - \frac{q_\mathrm{F} + q_\mathrm{G}}{\sharp\, r} - q_{\mathbb{D}^2_{sk}}(q_\mathrm{F} + 1)\left(\frac{1}{\sharp\, s} + \frac{1}{\sharp\, r}\right) - \frac{q_{\mathbb{D}^2_{sk}}}{\sharp\, k} - n_{max}\left(\frac{\nu}{2} + \frac{q_{\mathbb{D}^2_{sk}}}{\sharp\, k}\right) ,$$

within a time bounded by

$$\tau_\mathcal{B} = \tau + (q_{\mathbb{D}^2_{sk}} + 1)(q_\mathrm{F} + q_\mathrm{G})\, q_\mathrm{H} \cdot (\tau_\mathcal{E} + O(1)) ,$$

where $\tau_\mathcal{E}$ denotes the maximum time needed by $\mathcal{E}_{pk}$ for a single encryption.

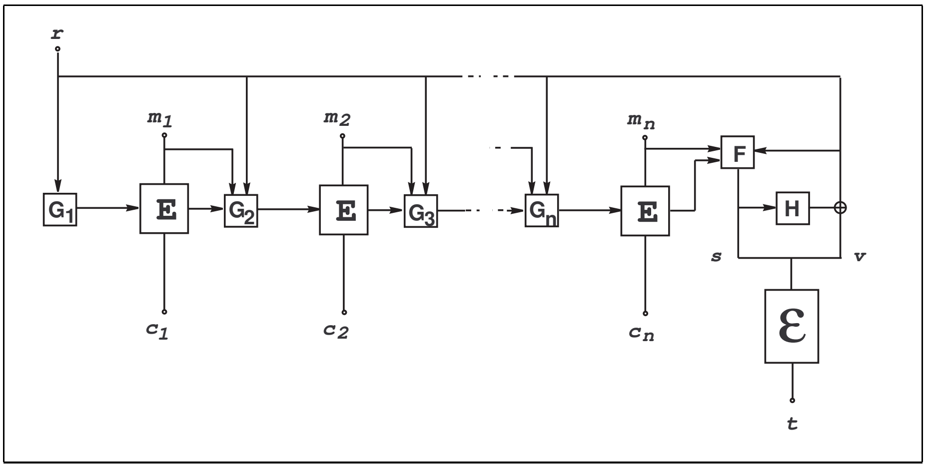

Fig. 2. Synopsis of GEM-2.

Again, the reader is invited to find the reduction proof in Appendix C for technical details.

4 Applications

Numerous applications are possible when embodying $\mathcal{E}_{pk}$ and $\mathtt{E}_k$. Due to lack of space, we will only consider the typical case $\mathcal{E}_{pk} = \mathrm{RSA}$ and $\mathtt{E}_k = \oplus$ (for which $\nu = 0$). The instantiations of random oracles F, H_i, H and G_i in one scheme or another by hash functions can be done by setting for instance $\mathrm{H}_i(\cdot) = \mathrm{SHA}(\cdot \| i)$ where the counter $i \in \langle 1, n_{max} \rangle$ is incremented at each block treatment. Special values of i such as 0 or -1 may be used to implement F and H.

4.1 $\mathbb{E}^{1}_{pk}[\mathbf{RSA}, \oplus]$

Corollary 1. *The encryption scheme* $\mathbb{E}^1_{pk}[\mathrm{RSA}, \oplus]$ *is* IND-CCA2 *in the random oracle model under the RSA assumption.*

For concrete security parameters, we suggest to use 1024-bit RSA keys with public exponent $e = \mathbb{F}_4 = 2^{16} + 1$. We set for instance $\log_2 \sharp\, t_2 = \mathsf{m} = \mathsf{k} = 160$ (hash functions F, H_i being derived from SHA-1 using a counter $i \in \langle 1, n_{max} \rangle$ like described above), $\sharp\, w = 2^{160}$ and $n_{max} = 2^{32}$. Assuming that the probability ε' to invert RSA lies around $\varepsilon' = 2^{-60}$, then an attaquer could distinguish $\mathbb{E}^1_{pk}[\mathrm{RSA}, \oplus]$ with $q_{\mathbb{D}^1_{sk}} = 2^{50}$ decryptions with advantage no more than $\varepsilon = 2^{-58}$.

From an implementation viewpoint, note that as soon as the RSA encryption has been done, the encryption procedure may directly output ciphertexts blocks one after the other without having to wait that all blocks are encrypted to transmit them all together. This allows on-the-fly encryption of communication streams. Three-tuples (w, y, k_1) may also be computed in advance to let the encryption device or software deal with hash computations only. The suggested setting allows to replace oracles $\mathrm{H}_2, \ldots, \mathrm{H}_n, \mathrm{F}$ by the compression function $(512 \mapsto 160)$ of SHA-1, driving us to $n+3$ calls to this function since the input of H_1 is made of three 512-bit blocks. Another benefit of our construction is that it requires only a small memory buffer (one field for the storage of w, one for the current key k_i and a third one for m_i). Finally, hardware implementations providing some hash coprocessor may drastically increase our speed rates.

4.2 $\mathbb{E}^2_{pk}[\mathbf{RSA}, \oplus]$

Corollary 2. *The encryption scheme* $\mathbb{E}^2_{pk}[\mathrm{RSA}, \oplus]$ *is* IND-CCA2 *in the random oracle model under the RSA assumption.*

For concrete security bounds, the same suggestions as previously lead to a maximal advantage of $\varepsilon = 2^{-58}$ if we take $\log_2 \sharp s = \log_2 \sharp r = 512$, $q_{\mathrm{F}} = q_{\mathrm{G}} = 2^{50}$ and $n_{max} = 2^{32}$.

Here again, any smart implementation allows on-the-fly encryption. The memory requirements are similar to the one of $\mathbb{E}^1_{pk}$. Here too, a coprocessor devoted to hash computations would increase speed rates.

5 Conclusion

We devised new generic constructions which apply to fixed-length weakly secure primitives and provide a strongly secure (IND-CCA2) public-key encryption scheme for messages of unfixed length like computer files or communications streams. An open question resides in investigating whether simpler and/or faster designs could exist, or whether the security requirements on the primitives could be shrunk further. Another challenging topic would be to come up with a construction holding only one additional field in the ciphertext but still employing a probabilistic encryption $\mathcal{E}_{pk}$ as in $\mathbb{E}^1_{pk}$. Finally, one could try to include a *signature* scheme in the encryption process to simultaneously authenticate the sender's identity, the plaintext and the ciphertext itself. Such an extension would ideally lead to fast and secure (according to one-more decryption attacks) signcryption schemes for arbitrary-length messages.

References

1. O. Baudron, D. Pointcheval, and J. Stern. Extended notions of security for multicast public key cryptosystems. In *Proc. of the 27th ICALP*, LNCS 1853, pages 499–511. Springer-Verlag, Berlin, 2000.
2. M. Bellare, A. Boldyreva, and S. Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In *Advances in Cryptology – EUROCRYPT '00*, LNCS 1807, pages 259–274. Springer-Verlag, Berlin, 2000.

3. Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations among notions of security for public-key encryption schemes. Full paper (30 pages), February 1999. An extended abstract appears in H. Krawczyk, ed., Advances in Cryptology – CRYPTO '98, volume 1462 of Lecture Notes in Computer Science, pages 26–45, Springer-Verlag, 1998.
4. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *First ACM Conference on Computer and Communications Security*, pages 62–73. ACM Press, 1993.
5. Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In A. De Santis, editor, *Advances in Cryptology – EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer-Verlag, 1995.
6. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
7. E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA–OAEP is secure under the rsa assumption. In *Advances in Cryptology – CRYPTO '01*, Lecture Notes in Computer Science. Springer-Verlag, 2001.
8. Eiichiro Fujisaki and Tatsuaki Okamoto. How to enhance the security of public-key encryption at minimum cost. In H. Imai and Y. Zheng, editors, *Public Key Cryptography*, volume 1560 of *Lecture Notes in Computer Science*, pages 53–68. Springer-Verlag, 1999.
9. Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In M. Wiener, editor, *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554. Springer-Verlag, 1999.
10. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
11. Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM Annual Symposium on the Theory of Computing (STOC '90)*, pages 427–437. ACM Press, 1990.
12. Tatsuaki Okamoto and David Pointcheval. The gap-problems: a new class of problems for the security of cryptographic schemes. In *PKC*, volume 1992 of *Lecture Notes in Computer Science*, pages 104–118. Springer-Verlag, 2001.
13. Tatsuaki Okamoto and David Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In D. Naccache, editor, *RSA 2001 Cryptographers' Track*, volume 2020 of *Lecture Notes in Computer Science*, pages 159–175. Springer-Verlag, 2001.
14. David Pointcheval. Chosen-ciphertext security for any one-way cryptosystem. In H. Imai and Y. Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 129–146. Springer-Verlag, 2000.
15. Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *Advances in Cryptology – CRYPTO '91*, volume 576, pages 433–444. Springer-Verlag, 1992.
16. Victor Shoup. Oaep reconsidered. In *Advances in Cryptology – CRYPTO '01*, Lecture Notes in Computer Science. Springer-Verlag, 2001.

A Preliminary

A.1 Notations

It is useful to introduce some notations. If a is some random variable, then $\sharp\, a$ denotes the number all possible values of a. For integers a and b, $\langle a, b\rangle$ denotes the set on integers ranging from a to b. For any predicate $\mathrm{R}(x)$, $\mathrm{R}(*)$ will stand for $\exists x$ s.t. $\mathrm{R}(x)$. If $\mathcal{O}$ is an oracle to which $\mathcal{A}$ has access, we denote by *query* $\mapsto$ *response* the correspondance $\mathcal{O}$ establishes between $\mathcal{A}$'s request *query* and the value *response* returned to $\mathcal{A}$. $\textsc{Hist}[\mathcal{O}]$ stands for the set of correspondances established by $\mathcal{O}$ as time goes on: $\textsc{Hist}[\mathcal{O}]$ can be seen as a memory which gets updated each time $\mathcal{A}$ makes a query to $\mathcal{O}$. We denote by $q_{\mathcal{O}}$ the number of calls $\mathcal{A}$ made to $\mathcal{O}$ during the simulation.

The relation $\mathsf{E}_1 \trianglelefteq \mathsf{E}_2$ indicates that the event E_1 takes place before the event E_2, if any of them occurs. In other words, when $\mathsf{E}_1 \trianglelefteq \mathsf{E}_2$ is true, if E_2 ever happens, then one knows for sure that E_1 happened before. By $\mathsf{E}_1, \ldots, \mathsf{E}_p \trianglelefteq \mathsf{E}'_1, \ldots, \mathsf{E}'_q$, we mean of course that $\mathsf{E}_i \trianglelefteq \mathsf{E}'_j$ stands for all $i = 1, p$ and $j = 1, q$. We note $\mathsf{E}_1 \triangleleft \mathsf{E}_2$ the event which sequentially realizes E_1 and then E_2. Equivalently, $\mathsf{E}_1 \triangleleft \mathsf{E}_2 = \mathsf{E}_1 \trianglelefteq \mathsf{E}_2 \wedge \mathsf{E}_2$. Again, $\mathsf{E}_1, \ldots, \mathsf{E}_p \triangleleft \mathsf{E}'_1, \ldots, \mathsf{E}'_q$, means $\mathsf{E}_i \triangleleft \mathsf{E}'_j$ for all $i = 1, p$ and $j = 1, q$. For convenience, $\odot \trianglelefteq \mathsf{E}_1$ (resp. $\mathsf{E}_1 \trianglelefteq \odot$) indicates that the event E_1 takes place during the guess (resp. find) stage of $\mathcal{A}$, $\odot$ representing the instants when $\mathcal{A}_1$ ends and $\mathcal{A}_2$ starts interchangeably.

A.2 Extending indistinguishability to scheme products

Let $\mathtt{E}^1$ and $\mathtt{E}^2$ be two symmetric encryption schemes. We define the scheme product of $\mathtt{E}^1$ and $\mathtt{E}^2$, $\mathtt{E} = \mathtt{E}^1 \times \mathtt{E}^2$ by

$$\mathtt{E}_k(m) = (\mathtt{E}^1 \times \mathtt{E}^2)_{(k_1,k_2)}(m_1, m_2) = (\mathtt{E}^1_{k_1}(m_1), \mathtt{E}^2_{k_2}(m_2)) ,$$

where all values stand in their respective sets. Then

Lemma 1. *If* $\mathtt{E}^1$ *is* (τ, ν_1)-IND *and* $\mathtt{E}^2$ *is* (τ, ν_2)-IND *then* E *is* $(\tau, \nu_1 + \nu_2)$-IND.

Proof. A proof of that fact will appear in the final version of this paper.

Note that [1] and [2] provide similar results for asymmetric encryption schemes. By immediate induction of lemma 1, we get that if $\mathtt{E}^i$ is (τ, ν_i)-IND for $i \in \langle 1, n \rangle$, then $\mathtt{E} = \prod_i \mathtt{E}^i$ is $(\tau, \sum_i \nu_i)$-IND. In particular, if $\mathtt{E}$ is (τ, ν)-IND, then $(\mathtt{E})^n$ is $(\tau, n\nu)$-IND.

B Security analysis of GEM-1

B.1 Description of the reduction algorithm

$\mathcal{B}$ is given an encryption $y = \mathcal{E}_{pk}(\widetilde{w}, *)$, an oracle $\mathcal{O}^{\mathrm{PCA}}$ which checks plaintexts for $\mathcal{E}_{pk}$, and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that breaks the $\mathsf{IND\text{-}CCA2}$ security of $\mathbb{E}^1_{pk}$. The goal of the reduction $\mathcal{B}$ is to retrieve the total knowledge of $\widetilde{w}$. Each time the reduction $\mathcal{B}$ needs to check whether a plaintext-ciphertext correspondance holds between y and w (which we denote $y = \mathcal{E}_{pk}(w, *)$), the query (y, w) is implicitly sent to $\mathcal{O}^{\mathrm{PCA}}$ which returns a boolean value. Wlog, we assume that $\mathcal{O}^{\mathrm{PCA}}$ responds to any of $\mathcal{B}$'s requests with no error and within a time bound τ_{PCA}.

Overview of $\mathcal{B}$. $\mathcal{B}$ runs $\mathcal{A}_1$ and provides a simulation for H_i with $i \in \langle 1, n_{max} \rangle$, F and $\mathbb{D}^1_{sk}$ as described later (find stage). $\mathcal{A}_1$ outputs a pair of message sequences (m^0, m^1) of identical blocklength $n \le n_{max}$ after a certain time. $\mathcal{B}$ then randomly chooses $b \in \{0, 1\}$ and proceeds to the following operations:

- if there exists $(w, y) \mapsto k_1 \in \textsc{Hist}[\mathrm{H}_1]$ with $y = \mathcal{E}_{pk}(w, *)$ then $\widetilde{w} := w$ and $\widetilde{k}_1 := k_1$ (event E_1) otherwise $\widetilde{k}_1$ is set to a random value,
- for $i \in \langle 2, n \rangle$, if there exists $(\widetilde{k}_{i-1}, m^b_i, w) \mapsto k_i \in \textsc{Hist}[\mathrm{H}_i]$ with $y = \mathcal{E}_{pk}(w, *)$ then $\widetilde{w} := w$ and $\widetilde{k}_i := k_i$ (event E_i); otherwise $\widetilde{k}_i$ is set to a random value,
- if there exists $(\widetilde{k}_n, m^b_n, w) \mapsto t_2 \in \textsc{Hist}[\mathrm{F}]$ with $y = \mathcal{E}_{pk}(w, *)$ then $\widetilde{w} := w$ and $\widetilde{t}_2 := t_2$ (event E_{F}); otherwise $\widetilde{t}_2$ is set to a random value.

$\mathcal{B}$ then computes $\widetilde{c}_i = \mathtt{E}_{\widetilde{k}_i}(m^b_i)$ for $i \in \langle 1, n \rangle$ and builds

$$\widetilde{c} = (y, \widetilde{c}_1, \ldots, \widetilde{c}_n, \widetilde{t}_2) \ .$$

This challenge is given to $\mathcal{A}_2$ which outputs some bit after another certain time (guess stage). Once finished, $\mathcal{B}$ will actually check whether some value $\widetilde{w}$ was defined during the game. If so, $\widetilde{w}$ is returned as the inversion of $\mathcal{E}_{pk}$ on y. Otherwise, the challenge y is simply rejected *i.e.* $\mathcal{B}$ sets $\widetilde{w} := \perp$ and stops. The simulation of random oracles as well as the simulation of the decryption oracle $\mathbb{D}^1_{sk}$ are detailed hereafter. Wlog, we assume that all simulated oracles keep tracks of their past queries throughout the game so that, if a query has been presented before and responded with some recorded output, then the same output is returned. In the sequel, all probabilities are taken over the random choices of $\mathcal{A}$ and $\mathcal{B}$ if not otherwise mentioned.

Simulation of H_1. For each new query (w, t_1),

(event E'_1) if $t_1 = y$ and $y = \mathcal{E}_{pk}(w, *)$ then H_1 sets $\widetilde{w} := w$, returns $\widetilde{k}_1$ and updates its history,

(event E''_1) else if $y = \mathcal{E}_{pk}(w, *)$ then H_1 sets $\widetilde{w} := w$, outputs a random value and updates its history,

(no event) else H_1 outputs a random value and updates its history.

Simulation of H_i for $i \in \langle 2, n \rangle$. For each new query (k, m, w),

(event E'_i) if processing guess stage and $k = \widetilde{k}_{i-1}$, $m = m^b_{i-1}$ and $y = \mathcal{E}_{pk}(w, *)$ then H_i sets $\widetilde{w} := w$, returns $\widetilde{k}_i$ and updates its history,

(event E''_i) else if $y = \mathcal{E}_{pk}(w, *)$ then H_i sets $\widetilde{w} := w$, outputs a random value and updates its history,

(no event) else H_i outputs a random value and updates its history.

Simulation of H_i for $i \in \langle n+1, n_{max} \rangle$. For each new query (k, m, w),

(event E_i) if $y = \mathcal{E}_{pk}(w, *)$ then H_i sets $\widetilde{w} := w$, outputs a random value and updates its history,

(no event) else H_i outputs a random value and updates its history.

Simulation of F. For each new query (k, m, w),

(event E'_{F}) if processing guess stage and $k = \widetilde{k}_n$, $m = m_n^b$ and $y = \mathcal{E}_{pk}(w, *)$ then F sets $\widetilde{w} := w$, returns $\widetilde{t_2}$ and updates its history,

(event $\mathsf{E}''_{\mathrm{F}}$) else if $y = \mathcal{E}_{pk}(w, *)$ then F sets $\widetilde{w} := w$, outputs a random value and updates its history,

(no event) else F outputs a random value and updates its history.

Simulation of $\mathbb{D}^1_{sk}$ (plaintext extractor). For each new query $(t_1, c_1, \ldots, c_d, t_2)$, $\mathbb{D}^1_{sk}$ first checks (this verification step only stands while the guess phase $\mathcal{A}_2$ is running) that $(t_1, c_1, \ldots, c_d, t_2) \neq (y, \widetilde{c}_1, \ldots, \widetilde{c}_n, \widetilde{t}_2)$ since if this equality holds, the query must be rejected as $\mathcal{A}$ attempts to decrypt its own challenge ciphertext. Then, $\mathbb{D}^1_{sk}$ tries to find the only (if any) message sequence $(m_1, \ldots, m_d)$ matching the query. To achieve this, $\mathbb{D}^1_{sk}$ invokes the simulations of the random oracles provided by $\mathcal{B}$ as follows:

- search for the unique $w \in \textsc{Hist}[\mathrm{H}_1] \cup \ldots \cup \textsc{Hist}[\mathrm{H}_d] \cup \textsc{Hist}[\mathrm{F}]$ such that $t_1 = \mathcal{E}_{pk}(w, *)$. If such a w exists,
  - query H_1 to get $k_1 = \mathrm{H}_1(w, t_1)$,
  - letting $m_1 = \mathsf{D}_{k_1}(c_1)$, query H_2 to get $k_2 = \mathrm{H}_2(k_1, m_1, w)$,
  - letting $m_2 = \mathsf{D}_{k_2}(c_2)$, query H_3 to get $k_3 = \mathrm{H}_3(k_2, m_2, w)$,

    $\vdots$
  - letting $m_{d-1} = \mathsf{D}_{k_{d-1}}(c_{d-1})$, query H_d to get $k_d = \mathrm{H}_d(k_{d-1}, m_{d-1}, w)$,
  - letting $m_d = \mathsf{D}_{k_d}(c_d)$, query F to check if $\mathrm{F}(k_d, m_d, w) = t_2$. If the equality holds, return $(m_1, \ldots, m_d)$; otherwise reject the query (event RJ_1).
- if the search for w is unsuccessful, reject the query (event RJ_2).

B.2 Soundness of $\mathcal{B}$

Simulation of random oracles.

Soundness of H_1. The simulation is perfect.

Soundness of H_i for $i \in \langle 2, n \rangle$. The simulation is perfect.

Soundness of H_i for $i \in \langle n+1, n_{max} \rangle$. The simulation is perfect.

Soundness of F. The simulation is perfect.

Plaintext extraction. The simulation of $\mathbb{D}^1_{sk}$ fails when $\perp$ is returned although the query $c = (t_1, c_1, \ldots, c_d, t_2)$ is a valid ciphertext. Let w and m_i, k_i for $i \in \langle 1, d\rangle$ denote the unique random variables associated to c in this case. Further define

$$\mathcal{H}_d = \bigcup_{i \in \langle 1, d\rangle} \textsc{Hist}\,[\mathrm{H}_i] \cup \textsc{Hist}\,[\mathrm{F}] \ .$$

Obviously, c was rejected through event RJ_2, because a rejection through RJ_1 refutes the validity of c. Therefore, if $\mathbb{D}^1_{sk}$ is incorrect for c, we must have

$$(\mathbb{D}^1_{sk} \text{ incorrect for } c) \wedge (c \text{ valid}) \quad \Rightarrow \quad w \notin \mathcal{H}_d \ .$$

We now decompose the failure event into several disjoint cases covering all possible situations.

Assume $(k_d \neq \widetilde{k}_n) \vee (m_d \neq m^b_n) \vee (w \neq \widetilde{w})$. Since $w \notin \textsc{Hist}\,[\mathrm{F}] \subset \mathcal{H}_d$, $\mathrm{F}(k_d, m_d, w)$ is a uniformly distributed random value unknown to $\mathcal{A}$. The fact that c is a valid ciphertext implies that $\mathrm{F}(k_d, m_d, w) = t_2$, which happens with probability

$$\Pr_{\mathrm{F}}\left[\mathrm{F}(k_d, m_d, w) = t_2\right] = \frac{1}{\sharp\, t_2} \ .$$

Assume $(k_d, m_d, w) = (\widetilde{k}_n, m^b_n, \widetilde{w})$ and $d > n$. Since $w \notin \textsc{Hist}\,[\mathrm{H}_d] \subset \mathcal{H}_d$, $\mathrm{H}_d(k_{d-1}, m_{d-1}, w)$ is a uniformly distributed random value unknown to $\mathcal{A}$. The fact that c is a valid ciphertext implies that $\mathrm{H}_d(k_{d-1}, m_{d-1}, w) = k_d = \widetilde{k}_n$, which happens with probability

$$\Pr_{\mathrm{H}_d}\left[\mathrm{H}_d(k_{d-1}, m_{d-1}, w) = \widetilde{k}_n\right] = \frac{1}{\sharp\, k} \ .$$

Assume $w = \widetilde{w}$, $d < n$ and $(k_i, m_i) = (\widetilde{k}_{n-d+i}, m^b_{n-d+i})$ for $i \in \langle 1, d\rangle$. If $t_1 = y$, we must have $\widetilde{k}_{n-d+1} = k_1 = \mathrm{H}_1(w, t_1) = \mathrm{H}_1(\widetilde{w}, y) = \widetilde{k}_1$ and this only happens with probability

$$\Pr_{\widetilde{k}_1, \widetilde{k}_{n-d+1}}\left[\widetilde{k}_1 = \widetilde{k}_{n-d+1}\right] = \frac{1}{\sharp\, k} \ .$$

Now suppose $t_1 \neq y$. This imposes $\mathrm{H}_1(w, t_1) = \widetilde{k}_{n-d+1}$. Because $w = \widetilde{w}$ was never queried to H_1, this situation occurs with probability

$$\Pr_{\mathrm{H}_1}\left[\mathrm{H}_1(w, t_1) = \widetilde{k}_{n-d+1}\right] = \frac{1}{\sharp\, k} \ .$$

ASSUME $w = \widetilde{w}$, $d = n$ AND $(k_i, m_i) = (\widetilde{k}_i, m_i^b)$ FOR $i \in \langle 1, n\rangle$. Obviously $t_1 \neq y$, since otherwise $c = \widetilde{c}$. Now we must have $\widetilde{k}_1 = k_1 = \mathrm{H}_1(w, t_1) = \mathrm{H}_1(\widetilde{w}, t_1)$, which happens with probability

$$\Pr_{\mathrm{H}_1}\left[\mathrm{H}_1(\widetilde{w}, t_1) = \widetilde{k}_1\right] = \frac{1}{\sharp\, k} .$$

ASSUME $w = \widetilde{w}$, $d \leq n$ AND $(k_i, m_i) \neq (\widetilde{k}_{n-d+i}, m_{n-d+i}^b)$ FOR SOME $i \in \langle 1, d{-}1\rangle$. Let us consider H_{j+1} where $j = \max_{i \leq d-1}\{(k_i, m_i) \neq (\widetilde{k}_{n-d+i}, m_{n-d+i}^b)\}$. We have $k_{j+1} = \mathrm{H}_{j+1}(k_j, m_j, w) = \widetilde{k}_{n-d+j+1}$, which, because w was never asked to H_{j+1}, occurs with probability

$$\Pr_{\mathrm{H}_{j+1}}\left[\mathrm{H}_{j+1}(k_j, m_j, w) = \widetilde{k}_{n-d+j+1}\right] = \frac{1}{\sharp\, k} .$$

CONCLUSION. Gathering all preceding bounds, we get

$$\Pr\left[c \text{ is valid} \wedge \mathbb{D}^1_{sk} \text{ incorrect for } c\right] \leq \frac{1}{\sharp\, t_2} + \frac{4}{\sharp\, k} + \sum_{j<d\leq n} \frac{1}{\sharp\, k} \leq \frac{1}{\sharp\, t_2} + \frac{n+3}{\sharp\, k} ,$$

which, taken over all queries of $\mathcal{A}_2$, leads to

$$\Pr\left[\mathbb{D}^1_{sk} \text{ incorrect}\right] \leq q_{\mathbb{D}^1_{sk}} \left(\frac{1}{\sharp\, t_2} + \frac{n+3}{\sharp\, k}\right) .$$

We further define $\dot{\Pr}[\cdot] = \Pr\left[\cdot \mid \neg(\mathbb{D}^1_{sk} \text{ incorrect})\right]$.

B.3 Reduction cost

Success probability. Let us suppose that $\mathcal{A}$ distinguishes $\mathbb{E}^1_{pk}$ within a time bound τ with advantage ε in less than q_{F}, $q_{\mathrm{H}} = \sum_{i\in\langle 1, n_{max}\rangle} q_{\mathrm{H}_i}$, $q_{\mathbb{D}^1_{sk}}$ oracle calls. This means that

$$\dot{\Pr}[\mathcal{A} = b] \geq \frac{1}{2} + \frac{\varepsilon}{2} .$$

Suppose also that $\mathtt{E}_k$ is (τ, ν)-indistinguishable. Assuming that the plaintext extractor is correctly simulated, if none of the events E_i, E'_i, E''_i or E_{F} occurs, then $\mathcal{A}$ never asked $\widetilde{w}$ to any of the random oracles and so could not learn any information whatsoever about the keys $\widetilde{k}_i$ under which the m_i^b were encrypted in $\widetilde{c}$ due to the randomness of the H_i. By virtue of lemma 1, this upper-limits the information leakage on b by $n\nu$, since $\mathcal{A}$'s running time is bounded by τ. Noting $\mathsf{E}_{win} = \mathsf{E}_{\mathrm{F}} \bigvee_{i\in\langle 1, n_{max}\rangle} \mathsf{E}_i \bigvee_{i\in\langle 1, n\rangle} \mathsf{E}'_i \vee \mathsf{E}''_i$, this means

$$\dot{\Pr}[\mathcal{A} = b \mid \neg\mathsf{E}_{win}] \leq \frac{1}{2} + \frac{n\nu}{2} .$$

We then get

$$\frac{1}{2} + \frac{\varepsilon}{2} \leq \dot{\Pr}\left[\mathcal{A} = b\right] \leq \dot{\Pr}\left[\mathcal{A} = b \mid \neg \mathsf{E}_{win}\right] + \dot{\Pr}\left[\mathsf{E}_{win}\right] \leq \frac{1}{2} + \frac{n\nu}{2} + \dot{\Pr}\left[\mathsf{E}_{win}\right] \ ,$$

wherefrom $\dot{\Pr}\left[\mathsf{E}_{win}\right] \geq (\varepsilon - n\nu)/2$. But $\dot{\Pr}\left[\mathcal{B} = \widetilde{w}\right] = \dot{\Pr}\left[\mathsf{E}_{win}\right]$ and finally,

$$\begin{aligned} \Pr\left[\mathcal{B} = \widetilde{w}\right] &\geq \dot{\Pr}\left[\mathcal{B} = \widetilde{w}\right] - \Pr\left[\mathbb{D}^1_{sk} \text{ incorrect}\right] \\ &\geq \frac{\varepsilon - n\nu}{2} - q_{\mathbb{D}^1_{sk}} \left(\frac{1}{\sharp\, t_2} + \frac{n+3}{\sharp\, k}\right) \ . \end{aligned}$$

Since the blocklength n of the message sequences (m^0, m^1) output by $\mathcal{A}_1$ cannot exceed n_{max}, $\mathcal{B}$ inverts $\mathcal{E}_{pk}$ on y with probability greater than

$$\frac{\varepsilon}{2} - q_{\mathbb{D}^1_{sk}} \left(\frac{1}{\sharp\, t_2} + \frac{3}{\sharp\, k}\right) - n_{max} \left(\frac{\nu}{2} + \frac{q_{\mathbb{D}^1_{sk}}}{\sharp\, k}\right) \ ,$$

i.e. succeeds with non-negligible probability.

Total number of calls to $\mathcal{O}^{\text{PCA}}$. Each simulated oracle H_i (resp. F) makes at most q_{H_i} (resp. q_{F}) queries to the plaintext-checking oracle. Note that the queries required by $\mathbb{D}^1_{sk}$ were already asked to $\mathcal{O}^{\text{PCA}}$ by either F or one of the H_i. By keeping tracks of all queries to $\mathcal{O}^{\text{PCA}}$, it is easy to see that the total number of calls actually needed by $\mathcal{B}$ is upper-bounded by

$$q_{\mathcal{O}^{\text{PCA}}} \leq q_{\mathrm{F}} + q_{\mathrm{H}} \quad \text{where} \quad q_{\mathrm{H}} = \sum_{i \in \langle 1, n_{max} \rangle} q_{\mathrm{H}_i} \ .$$

Total running time. The reduction algorithm runs in time bounded by

$$\tau_{\mathcal{B}} = \tau + (q_{\mathbb{D}^1_{sk}} + 1)\,(q_{\mathrm{F}} + q_{\mathrm{H}}) \cdot (\tau_{\text{PCA}} + O(1)) \ .$$

C Security Analysis of GEM-2

C.1 Description of the reduction algorithm

$\mathcal{B}$ is given an encryption $y = \mathcal{E}_{pk}(\widetilde{w})$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that breaks the IND-CCA2 security of $\mathbb{E}^2_{pk}$. The goal of the reduction $\mathcal{B}$ is to retrieve the total knowledge of $\widetilde{w}$.

Overview of $\mathcal{B}$. $\mathcal{B}$ runs $\mathcal{A}_1$ and provides a simulation for G_i with $i \in \langle 1, n_{max} \rangle$, F, H and $\mathbb{D}^2_{sk}$ as described later (find stage). $\mathcal{A}_1$ outputs a pair of message sequences (m^0, m^1) of identical blocklength $n \leq n_{max}$ after a certain time. $\mathcal{B}$ then chooses $b \in \{0,1\}$, $\widetilde{k}_1, \ldots, \widetilde{k}_n$ uniformly at random, computes $\widetilde{c}_i = \mathtt{E}_{\widetilde{k}_i}(m^b_i)$ for $i \in \langle 1, n \rangle$ and builds

$$\widetilde{c} = (\widetilde{c}_1, \ldots, \widetilde{c}_n, y) \ .$$

This challenge is given to $\mathcal{A}_2$ which outputs some bit after another certain time (guess stage). Once finished, $\mathcal{B}$ will actually check whether some value $\widetilde{w}$ was defined during the game. If so, $\widetilde{w}$ is returned as the inversion of $\mathcal{E}_{pk}$ on y. Otherwise, the challenge y is simply rejected *i.e.* $\mathcal{B}$ sets $\widetilde{w} := \perp$ and stops. The simulation of random oracles as well as the simulation of the decryption oracle $\mathbb{D}^2_{sk}$ are detailed hereafter. Wlog, we assume that all simulated oracles keep tracks of their past queries throughout the game so that, if a query has been presented before and responded with some recorded output, then the same output is returned. In the sequel, all probabilities are taken over the random choices of $\mathcal{A}$ and $\mathcal{B}$ if not otherwise mentioned.

Simulation of G_1. For each new query r,

(event E_1) if processing guess stage and there exists $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ such that $y = \mathcal{E}_{pk}(s \| r \oplus h)$ then G_1 sets $\widetilde{w} := s \| r \oplus h$, returns $\widetilde{k}_1$ and updates its history,

(event E'_1) else if there exists $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ such that $y = \mathcal{E}_{pk}(s \| r \oplus h)$ then G_1 sets $\widetilde{w} := s \| r \oplus h$, outputs a random value and updates its history,

(no event) else G_1 outputs a random value and updates its history.

Simulation of G_i for $i \in \langle 2, n \rangle$. For each new query (k, m, r),

(event E_i) if processing guess stage and $k = \widetilde{k}_i$, $m = m^b_{i-1}$ and there exists $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ such that $y = \mathcal{E}_{pk}(s \| r \oplus h)$ then G_i sets $\widetilde{w} := s \| r \oplus h$, returns $\widetilde{k}_i$ and updates its history,

(event E'_i) else if there exists $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ such that $y = \mathcal{E}_{pk}(s \| r \oplus h)$ then G_i sets $\widetilde{w} := s \| r \oplus h$, outputs a random value and updates its history,

(no event) else G_i outputs a random value and updates its history.

Simulation of G_i for $i \in \langle n+1, n_{max} \rangle$. For each new query (k, m, r),

(event E_i) if there exists $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ such that $y = \mathcal{E}_{pk}(s \| r \oplus h)$ then G_i sets $\widetilde{w} := s \| r \oplus h$, outputs a random value and updates its history,

(no event) else G_i outputs a random value and updates its history.

Simulation of F. For each new query (k, m, r),

(event E_{F}) if processing guess stage and $k = \widetilde{k}_n$, $m = m_n^b$ and there exists $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ such that $y = \mathcal{E}_{pk}(s\|r \oplus h)$ then F sets $\widetilde{w} := s\|r \oplus h$, returns s and updates its history,

(event E'_{F}) if there exists $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ such that $y = \mathcal{E}_{pk}(s\|r \oplus h)$ then F sets $\widetilde{w} := s\|r \oplus h$, outputs a random value and updates its history,

(no event) else F outputs a random value and updates its history.

Simulation of H. For each new query s, H outputs a random value and updates its history.

Simulation of $\mathbb{D}^2_{sk}$ (plaintext extractor). For each new query $(c_1, \ldots, c_d, t)$, $\mathbb{D}^2_{sk}$ first checks (this verification step only stands while the guess stage $\mathcal{A}_2$ is running) that $(c_1, \ldots, c_d, t) \neq (\widetilde{c_1}, \ldots, \widetilde{c_n}, y)$ since if this equality holds, the query must be rejected as $\mathcal{A}$ attempts to decrypt its own challenge ciphertext. Then, $\mathbb{D}^2_{sk}$ attempts to find the only (if any) message sequence $(m_1, \ldots, m_d)$ matching the query. To achieve this, $\mathbb{D}^2_{sk}$ invokes the simulations of the random oracles provided by $\mathcal{B}$ as follows:

- search for the unique pair (r, s) such that $r \in \textsc{Hist}[\mathrm{G}_1] \cup \ldots \cup \textsc{Hist}[\mathrm{G}_d] \cup \textsc{Hist}[\mathrm{F}]$, $s \mapsto h \in \textsc{Hist}[\mathrm{H}]$ and $t = \mathcal{E}_{pk}(s\|r \oplus h)$. If such a pair exists,
  - query G_1 to get $k_1 = \mathrm{G}_1(r)$,
  - letting $m_1 = \mathtt{D}_{k_1}(c_1)$, query G_2 to get $k_2 = \mathrm{G}_2(k_1, m_1, r)$,
  - letting $m_2 = \mathtt{D}_{k_2}(c_2)$, query G_3 to get $k_3 = \mathrm{H}_3(k_2, m_2, r)$,

    $\vdots$
  - letting $m_{d-1} = \mathtt{D}_{k_{d-1}}(c_{d-1})$, query G_d to get $k_d = \mathrm{G}_d(k_{d-1}, m_{d-1}, r)$,
  - letting $m_d = \mathtt{D}_{k_d}(c_d)$, query F to check if $\mathrm{F}(k_d, m_d, r) = s$. If the equality holds, return $(m_1, \ldots, m_d)$; otherwise reject the query (event RJ_1).
- if the search for (r, s) is unsuccessful, reject the query (event RJ_2).

C.2 Soundness of $\mathcal{B}$

Simulation of random oracles. The plaintext $\widetilde{w}$ uniquely defines $\widetilde{s}$ and $\widetilde{v}$ such that $\widetilde{w} = \widetilde{s}\|\widetilde{v}$. We note $\widetilde{r}$ the random variable $\widetilde{v} \oplus \mathrm{H}(\widetilde{s})$. We denote by

- $\mathsf{E}_{\widetilde{s}}$ the event that $\mathcal{A}$ queries $\widetilde{s}$ to the oracle H,
- $\mathsf{E}_{\mathrm{G}_1}$ the event that $\mathcal{A}$ queries $\widetilde{r}$ to G_1,
- for $i \in \langle 2, n \rangle$, $\mathsf{E}_{\mathrm{G}_i}$ the event that $\mathcal{A}$ queries $(\widetilde{k}_{i-1}, m_{i-1}^b, \widetilde{r})$ to G_i,
- E_{F} the event that $\mathcal{A}$ queries $(\widetilde{k}_n, m_n^b, \widetilde{r})$ to F,
- $\mathsf{E}_{\widetilde{r}}$ the event that $\mathcal{A}$ queries $\widetilde{r}$ to any of the oracles F, G_i *i.e.* $\mathsf{E}_{\mathrm{F}} \vee \mathsf{E}_{\mathrm{G}_1} \vee \ldots \vee \mathsf{E}_{\mathrm{G}_n}$.

Soundness of G_i for $i \in \langle 1, n \rangle$. The simulation of G_i fails when $(\widetilde{k}_{i-1}, m^b_{i-1}, \widetilde{r})$, or $\widetilde{r}$ in the case of G_1, is queried and answered with some value $k_i \neq \widetilde{k}_i$ before $\widetilde{s}$ appears in Hist [H]. More precisely, the simulation is perfect if and only if the predicate $(\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\mathrm{G}_i})$ is fulfilled, which yields $(\mathrm{G}_i \text{ incorrect}) \Leftrightarrow \neg(\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\mathrm{G}_i})$.

Soundness of G_i for $i \in \langle n+1, n_{max} \rangle$. The simulation is perfect.

Soundness of F. The simulation of F fails when $(\widetilde{k}_n, m^b_n, \widetilde{r})$ is queried and answered with some value $s \neq \widetilde{s}$ before $\widetilde{s}$ appears in Hist [H]. Here, the simulation runs perfectly if and only if $\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\mathrm{F}}$. Hence, $(\mathrm{F} \text{ incorrect}) \Leftrightarrow \neg(\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\mathrm{F}})$.

Soundness of H. The simulation is perfect.

Conclusion. Gathering preceding results, using $\neg(\mathsf{E}_1 \trianglelefteq \mathsf{E}_2) = (\neg\mathsf{E}_1 \wedge \mathsf{E}_2) \vee (\mathsf{E}_2 \triangleleft \mathsf{E}_1)$ and reorganizing in disjoint events, one gets

$$\begin{aligned} \text{incorrect oracle} \quad &\Leftrightarrow \quad \vee_{i \leq n} \neg(\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\mathrm{G}_i}) \vee \neg(\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\mathrm{F}}) \\ &\Leftrightarrow \quad \neg(\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq (\vee_{i \in \langle 1, n \rangle} \mathsf{E}_{\mathrm{G}_i} \vee \mathsf{E}_{\mathrm{F}})) \\ &\Leftrightarrow \quad \neg(\odot, \mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\widetilde{r}}) \\ &\Leftrightarrow \quad (\odot \trianglelefteq \mathsf{E}_{\widetilde{r}}) \wedge \neg(\mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\widetilde{r}}) \vee (\mathsf{E}_{\widetilde{r}} \triangleleft \odot) \,, \end{aligned}$$

wherefrom

$$\begin{aligned} \Pr[\text{incorrect oracle}] &= \Pr[(\odot \trianglelefteq \mathsf{E}_{\widetilde{r}}) \wedge \neg(\mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\widetilde{r}})] + \Pr[\mathsf{E}_{\widetilde{r}} \triangleleft \odot] \\ &\leq \Pr[\odot \trianglelefteq \mathsf{E}_{\widetilde{r}} \mid \neg(\mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\widetilde{r}})] + \Pr[\mathsf{E}_{\widetilde{r}} \triangleleft \odot] \ . \end{aligned}$$

Since $\mathcal{A}_1$ does not have access to y and because y has a uniform distribution, $\widetilde{r}$ is a uniformly distributed random variable throughout the find stage. Hence

$$\Pr[\mathsf{E}_{\widetilde{r}} \triangleleft \odot] \leq \frac{q^1_{\mathrm{F}} + \sum_{i \leq n} q^1_{\mathrm{G}_i}}{\sharp r} \,,$$

where $q^1_{\mathcal{O}}$ is the number of calls to oracles $\mathcal{O} \in \{\mathrm{G}_1, \ldots, \mathrm{G}_n, \mathrm{F}\}$ that $\mathcal{A}_1$ made during the find stage. Now, throughout the guess stage, $\mathcal{A}_2$ cannot gain any information about $\widetilde{r} = \widetilde{v} \oplus \mathrm{H}(\widetilde{s})$ without knowing $\mathrm{H}(\widetilde{s})$ *i.e.* without submitting $\widetilde{s}$ to H. Hence,

$$\Pr[\odot \trianglelefteq \mathsf{E}_{\widetilde{r}} \mid \neg(\mathsf{E}_{\widetilde{s}} \trianglelefteq \mathsf{E}_{\widetilde{r}})] \leq \frac{q^2_{\mathrm{F}} + \sum_{i \leq n} q^2_{\mathrm{G}_i}}{\sharp r} \ .$$

Finally, the probability that an error occurs while $\mathcal{B}$ simulates the oracles F, $\mathrm{G}_1, \ldots, \mathrm{G}_n$ is upper-bounded by

$$\Pr[\text{incorrect oracle}] \leq \frac{q_{\mathrm{F}} + \sum_{i \leq n} q_{\mathrm{G}_i}}{\sharp r} \ .$$

We further define $\dot{\Pr}[\cdot] = \Pr[\cdot \mid \neg(\text{incorrect oracle})]$.

Plaintext extraction. Assume that all random oracles are perfectly simulated throughout the game. The simulation of $\mathbb{D}^2_{sk}$ fails when $\perp$ is returned although the query $c = (c_1, \ldots, c_d, t)$ is a valid ciphertext. Let r, s, v and m_i, k_i for $i \in \langle 1, d\rangle$ denote the unique random variables associated to c in this case. Further define

$$\mathcal{G}_d = \bigcup_{i \in \langle 1,d\rangle} \textsc{Hist}\,[\mathrm{G}_i] \cup \textsc{Hist}\,[\mathrm{F}] \ .$$

Obviously, c was rejected through event RJ_2, because a rejection through RJ_1 refutes the validity of c. Therefore, if $\mathbb{D}^2_{sk}$ is incorrect for c, we must have

$$(\mathbb{D}^2_{sk} \text{ incorrect for } c) \wedge (c \text{ valid}) \quad \Rightarrow \quad r \notin \mathcal{G}_d \vee s \notin \textsc{Hist}\,[\mathrm{H}] \ .$$

We now decompose the failure event into several disjoint cases covering all possible situations.

Assume $(k_d, m_d, r) \neq (\widetilde{k}_n, m^b_n, \widetilde{r})$ and $r \notin \mathcal{G}_d$. Since $r \notin \textsc{Hist}\,[\mathrm{F}] \subset \mathcal{H}_d$, $\mathrm{F}(k_d, m_d, r)$ is a uniformly distributed random value unknown to $\mathcal{A}$. The fact that c is a valid ciphertext implies that $\mathrm{F}(k_d, m_d, r) = s$, which happens with probability

$$\dot{\Pr_{\mathrm{F}}}\ [\mathrm{F}(k_d, m_d, r) = s] = \frac{1}{\sharp\, s} \ .$$

Assume $(k_d, m_d, r) \neq (\widetilde{k}_n, m^b_n, \widetilde{r})$ and $r \in \textsc{Hist}\,[\mathrm{F}] \wedge s \notin \textsc{Hist}\,[\mathrm{H}]$. Suppose that $s \neq \widetilde{s}$. Since $s \notin \textsc{Hist}\,[\mathrm{H}]$, $\mathrm{H}(s)$ is a uniformly distributed random value unknown to $\mathcal{A}$. The validity of c implies that $(k_d, m_d, v \oplus \mathrm{H}(s)) \mapsto s \in \textsc{Hist}\,[\mathrm{F}]$, which happens with probability

$$\dot{\Pr_{\mathrm{H}}}\ [(k_d, m_d, v \oplus \mathrm{H}(s)) \mapsto s \in \textsc{Hist}\,[\mathrm{F}]] \leq \frac{q_{\mathrm{F}}}{\sharp\, r} \ .$$

Now assume $s = \widetilde{s}$. In this case, we must have $(k_d, m_d, r) \mapsto \widetilde{s} \in \textsc{Hist}\,[\mathrm{F}]$ which occurs with probability

$$\dot{\Pr_{\mathrm{F}}}\ [(k_d, m_d, r) \mapsto \widetilde{s} \in \textsc{Hist}\,[\mathrm{F}]] \leq \frac{q_{\mathrm{F}}}{\sharp\, s} \ .$$

Assume $(k_d, m_d, r) = (\widetilde{k}_n, m^b_n, \widetilde{r})$ and $s \neq \widetilde{s}$. This is absurd since $s = \mathrm{F}(k_d, m_d, r) = \mathrm{F}(\widetilde{k}_n, m^b_n, \widetilde{r}) = \widetilde{s}$.

Assume $(k_d, m_d, r, s) = (\widetilde{k}_n, m^b_n, \widetilde{r}, \widetilde{s})$ and $\widetilde{s} \notin \textsc{Hist}\,[\mathrm{H}]$. Since c is a valid ciphertext, we must have $r = v \oplus \mathrm{H}(s)$ *i.e.* $\mathrm{H}(\widetilde{s}) = \widetilde{r} \oplus v$. Because $\widetilde{s}$ was never given to H, this happens with probability

$$\dot{\Pr_{\mathrm{H}}}\ [\mathrm{H}(\widetilde{s}) = \widetilde{r} \oplus v] = \frac{1}{\sharp\, r} \ .$$

From now on, we suppose that $r = \widetilde{r}$, $s = \widetilde{s}$, $\widetilde{s} \mapsto \widetilde{r} \oplus \widetilde{v} \in \textsc{Hist}\,[\mathrm{H}]$ (so that $t = y$) and $\widetilde{r} \notin \mathcal{G}_d$.

Assume $(k_d, m_d) = (\widetilde{k}_n, m_n^b)$ and $d > n$. Since $r \notin \textsc{Hist}[\mathrm{G}_d]$, $\mathrm{G}_d(k_{d-1}, m_{d-1}, r)$ is a uniformly distributed random value unknown to $\mathcal{A}$. The fact that c is a valid ciphertext implies that $\mathrm{G}_d(k_{d-1}, m_{d-1}, r) = k_d = \widetilde{k}_n$, which happens with probability

$$\dot{\Pr_{\mathrm{G}_d}}\left[\mathrm{G}_d(k_{d-1}, m_{d-1}, r) = \widetilde{k}_n\right] = \frac{1}{\sharp\, k} \ .$$

Assume $d \leq n$ and $(k_i, m_i) = (\widetilde{k}_{n-d+i}, m_{n-d+i}^b)$ for $i \in \langle 1, d \rangle$. The case $d = n$ leads to the absurd equality $c = \widetilde{c}$. Suppose $d < n$. Considering G_1, we must have $\widetilde{k}_{n-d+1} = k_1 = \mathrm{H}_1(r) = \mathrm{H}_1(\widetilde{r}) = \widetilde{k}_1$ and this only happens with probability

$$\dot{\Pr_{\widetilde{k}_1, \widetilde{k}_{n-d+1}}}\left[\widetilde{k}_1 = \widetilde{k}_{n-d+1}\right] = \frac{1}{\sharp\, k} \ .$$

Assume $d \leq n$ and $(k_i, m_i) \neq (\widetilde{k}_{n-d+i}, m_{n-d+i}^b)$ for some $i \in \langle 1, d-1 \rangle$. Let us consider G_{j+1} where $j = \max_{i \leq d-1}\{(k_i, m_i) \neq (\widetilde{k}_{n-d+i}, m_{n-d+i}^b)\}$. We have $k_{j+1} = \mathrm{G}_{j+1}(k_j, m_j, \widetilde{r}) = \widetilde{k}_{n-d+j+1}$, which, because $\widetilde{r}$ was never asked to H_{j+1}, occurs with probability

$$\dot{\Pr_{\mathrm{G}_{j+1}}}\left[\mathrm{G}_{j+1}(k_j, m_j, \widetilde{r}) = \widetilde{k}_{n-d+j+1}\right] = \frac{1}{\sharp\, k} \ .$$

Conclusion. Gathering all preceding bounds, we get

$$\begin{aligned}
\dot{\Pr}\left[c \text{ is valid} \wedge \mathbb{D}_{sk}^2 \text{ incorrect for } c\right] &\leq (q_{\mathrm{F}} + 1)\left(\frac{1}{\sharp\, s} + \frac{1}{\sharp\, r}\right) + \frac{1}{\sharp\, k} + \sum_{j < d \leq n} \frac{1}{\sharp\, k} \\
&\leq (q_{\mathrm{F}} + 1)\left(\frac{1}{\sharp\, s} + \frac{1}{\sharp\, r}\right) + \frac{n+1}{\sharp\, k} \ ,
\end{aligned}$$

which, taken over all queries of $\mathcal{A}_2$, leads to

$$\Pr\left[\mathbb{D}_{sk}^2 \text{ incorrect}\right] \leq q_{\mathbb{D}_{sk}^2}\left((q_{\mathrm{F}} + 1)\left(\frac{1}{\sharp\, s} + \frac{1}{\sharp\, r}\right) + \frac{n+1}{\sharp\, k}\right) \ .$$

We have

$$\begin{aligned}
\Pr\left[\mathcal{B} \text{ incorrect}\right] &= \Pr\left[\text{incorrect oracle}\right] + \Pr\left[\mathbb{D}_{sk}^2 \text{ incorrect} \wedge \neg(\text{incorrect oracle})\right] \\
&\leq \Pr\left[\text{incorrect oracle}\right] + \Pr\left[\mathbb{D}_{sk}^2 \text{ incorrect} \mid \neg(\text{incorrect oracle})\right] \\
&\leq \frac{q_{\mathrm{F}} + \sum_i q_{\mathrm{G}_i}}{\sharp\, r} + q_{\mathbb{D}_{sk}^2}\left((q_{\mathrm{F}} + 1)\left(\frac{1}{\sharp\, s} + \frac{1}{\sharp\, r}\right) + \frac{n+1}{\sharp\, k}\right) \ .
\end{aligned}$$

We further define $\ddot{\Pr}[\cdot] = \Pr[\cdot \mid \neg(\mathcal{B} \text{ incorrect})]$.

C.3 Reduction cost

Success probability. Let us suppose that $\mathcal{A}$ distinguishes $\mathbb{E}^2_{pk}$ within a time bound τ with advantage ε in less than q_{F}, q_{H}, $q_{\mathrm{G}} = \sum_{i \in \langle 1, n_{max} \rangle} q_{\mathrm{G}_i}$, $q_{\mathbb{D}^2_{sk}}$ oracle calls. This means that

$$\ddot{\Pr}\left[\mathcal{A} = b\right] \geq \frac{1}{2} + \frac{\varepsilon}{2} .$$

Suppose also that $\mathtt{E}_k$ is (τ, ν)-indistinguishable. Assuming that the random oracles and the plaintext extractor are perfectly simulated, if none of the events E_i, E'_i or E_{F} occurs, then $\mathcal{A}$ never asked $\widetilde{r}$ to any of the random oracles and so could not learn any information whatsoever about the keys $\widetilde{k}_i$ under which the m^b_i were encrypted in $\widetilde{c}$ due to the randomness of the G_i. By virtue of lemma 1, this upper-limits the information leakage on b by $n\nu$, since $\mathcal{A}$'s running time is bounded by τ. Noting $\mathsf{E}_{win} = \mathsf{E}_{\mathrm{F}} \bigvee_{i \in \langle 1, n_{max} \rangle} \mathsf{E}_i \bigvee_{i \in \langle 1, n \rangle} \mathsf{E}'_i$, this means

$$\ddot{\Pr}\left[\mathcal{A} = b \mid \neg \mathsf{E}_{win}\right] \leq \frac{1}{2} + \frac{n\nu}{2} .$$

We then get

$$\frac{1}{2} + \frac{\varepsilon}{2} \leq \ddot{\Pr}\left[\mathcal{A} = b\right] \leq \ddot{\Pr}\left[\mathcal{A} = b \mid \neg \mathsf{E}_{win}\right] + \ddot{\Pr}\left[\mathsf{E}_{win}\right] \leq \frac{1}{2} + \frac{n\nu}{2} + \ddot{\Pr}\left[\mathsf{E}_{win}\right] ,$$

wherefrom $\ddot{\Pr}\left[\mathsf{E}_{win}\right] \geq (\varepsilon - n\nu)/2$. But $\ddot{\Pr}\left[\mathcal{B} = \widetilde{w}\right] = \ddot{\Pr}\left[\mathsf{E}_{win}\right]$ and finally,

$$\begin{aligned} \Pr\left[\mathcal{B} = \widetilde{w}\right] &\geq \ddot{\Pr}\left[\mathcal{B} = \widetilde{w}\right] - \Pr\left[\mathbb{D}^2_{sk} \text{ incorrect}\right] \\ &\geq \frac{\varepsilon - n\nu}{2} - \frac{q_{\mathrm{F}} + \sum_i q_{\mathrm{G}_i}}{\sharp\, r} - q_{\mathbb{D}^2_{sk}} \left((q_{\mathrm{F}} + 1) \left(\frac{1}{\sharp\, s} + \frac{1}{\sharp\, r} \right) + \frac{n+1}{\sharp\, k} \right) . \end{aligned}$$

Since the blocklength n of the message sequences (m^0, m^1) output by $\mathcal{A}_1$ cannot exceed n_{max}, $\mathcal{B}$ inverts $\mathcal{E}_{pk}$ on y with probability greater than

$$\frac{\varepsilon}{2} - \frac{q_{\mathrm{F}} + q_{\mathrm{G}}}{\sharp\, r} - q_{\mathbb{D}^2_{sk}} (q_{\mathrm{F}} + 1) \left(\frac{1}{\sharp\, s} + \frac{1}{\sharp\, r} \right) - \frac{q_{\mathbb{D}^2_{sk}}}{\sharp\, k} - n_{max} \left(\frac{\nu}{2} + \frac{q_{\mathbb{D}^2_{sk}}}{\sharp\, k} \right) ,$$

i.e. succeeds with non-negligible probability.

Total running time. The reduction algorithm runs in time bounded by

$$\tau_{\mathcal{B}} = \tau + (q_{\mathbb{D}^2_{sk}} + 1)\,(q_{\mathrm{F}} + q_{\mathrm{G}})\, q_{\mathrm{H}} \cdot (\tau_{\mathcal{E}} + O(1)) ,$$

where $\tau_{\mathcal{E}}$ denotes the maximum time needed by $\mathcal{E}_{pk}$ for a single encryption.