A CCA Secure Hybrid Damgård's ElGamal Encryption

Yvo Desmedt[1]* and Duong Hieu Phan[2] **

[1] University College London
Gower Street, London WC1E 6BT, United Kingdom.
[2] University of Paris 8
2, rue de la Liberté 93526 - Saint-Denis cedex 02, France
`hieu.phan@univ-paris8.fr`

Abstract. ElGamal encryption, by its efficiency, is one of the most used schemes in cryptographic applications. However, the original ElGamal scheme is only provably secure against passive attacks. Damgård proposed a slight modification of ElGamal encryption scheme (named Damgård's ElGamal scheme) that provides security against non-adaptive chosen ciphertext attacks under a knowledge-of-exponent assumption. Recently, the CCA1-security of Damgård's ElGamal scheme has been proven under more standard assumptions.
In this paper, we study the open problem of CCA2-security of Damgård's ElGamal. By employing a data encapsulation mechanism, we prove that the resulted hybrid Damgård's ElGamal Encryption is secure against adaptive chosen ciphertext attacks. The down side is that the proof of security is based on a knowledge-of-exponent assumption. In terms of efficiency, this scheme is more efficient (e.g. one exponentiation less in encryption) than Kurosawa-Desmedt scheme, the most efficient scheme in the standard model so far.

1 Introduction

ElGamal encryption was introduced by ElGamal in 1985 [7] as a variant of Diffie-Hellman key exchange. Since then, it has become one of the two most extensively used for cryptographic applications, besides RSA [17].

The notion of *semantic security*, introduced by Goldwasser and Micali [11], captures the intuition that an adversary should not be able to obtain any partial information about the underlying plaintext of a *challenge* ciphertext. At the same time, various kinds of attack have been modeled and the strongest formalized attack is a chosen ciphertext attack where the adversary is given access to a *decryption oracle* that allows him to obtain the decryptions of his chosen

* The author is BT Professor of Information Security and is also funded by EPSRC EP/C538285/1.

** Part of this research was done while being at University College London, Adastral Campus.

ciphertexts (with a natural restriction that these chosen ciphertexts are different from the challenge ciphertext). There are two kinds of chosen ciphertext attack: *non-adaptive attacks* or *lunchtime attacks* [15], denoted $\mathsf{CCA1}$, where the access to the decryption oracle is limited until the challenge is known; and *adaptive chosen-ciphertext attack* [16], denoted $\mathsf{CCA2}$ or CCA where adversaries have access to decryption oracle before and after receiving the challenge.

The original ElGamal scheme was proven to be semantically secure against passive adversaries and there was the open question whether it is secure against $\mathsf{CCA1}$ attacks.

In [5] Damgård proposed a variant of the ElGamal encryption scheme (later called Damgård's ElGamal Encryption), by only adding an exponentiation to ciphertexts, and provided a proof of security against non-adaptive chosen ciphertext attacks. This proof requires however an informal assumption of *knowledge-of-exponent* assumption. Later, Bellare and Palacio [2] formalized this notion, calling it the *Diffie-Hellman knowledge* (DHK for short) assumption, and provided a formal proof of security against non-adaptive chosen ciphertext attacks for Damgård's ElGamal Encryption. Dent [6] has shown that the DHK assumption holds in generic groups. Recently, Gjøsteen [10], Wu and Stinson [21], Lipmaa [14] improved the security results of Damgård's ElGamal Encryption but all of these works only concern $\mathsf{CCA1}$ security.

The Damgård's ElGamal encryption is obviously not CCA secure because it is homomorphic. An important problem is thus to consider a simple variant of Damgård's ElGamal encryption that could achieve CCA security, as this security level is well-admitted to be the standard notion for confidentiality in cryptography. We investigate in this paper the natural issue of using Damgård's ElGamal encryption in a hybrid framework.

A hybrid encryption scheme [20] employs public-key encryption techniques to derive a shared key that is then used to encrypt the actual messages using symmetric-key techniques. It can thus take advantage of symmetric encryption to encrypt arbitrary long messages. In [4], a formal treatment of hybrid schemes was proposed, composed of two parts : first, a KEM (Key Encapsulation Mechanism) is invoked to encrypt a random key; second, a DEM (Data Encapsulation Mechanism) is performed to encrypt messages using a symmetric encryption scheme. Kurosawa-Desmedt [13] proposed later a hybrid variant of the Cramer-Shoup scheme [3] in the KEM/DEM framework which results in the most efficient scheme in the standard model so far.

Contribution. We propose a hybrid Damgård's ElGamal Encryption, *i.e.*, embedding a DEM in the Damgård's ElGamal Encryption, and show that this scheme achieves CCA security. The proposed scheme is more efficient than the Kurosawa-Desmedt scheme in many aspects: it requires one less exponentiation, it needs no target collision resistance, it produces shorter secret keys and public keys.

The security proof for our proposed hybrid Damgård's ElGamal scheme has to be however based on an extension of DHK assumption (which is an another formalization of the $\mathsf{KEA3}$ assumption in [1]). We give a proof of the hardness of this assumption in generic groups. Although this kind of proof in generic groups can be seen as a minimum requirement for the use of an assumption in a security proof, we strongly believe that the hybrid Damgård's ElGamal scheme would be very useful in practice and deserves to be investigated further.

2 Notation and standard definitions

In this section, we recall the formalization of the most important security notion for asymmetric encryption, namely the CCA security.

Firstly, let us briefly remind that a public-key encryption scheme π is defined by three algorithms: the key generation algorithm $\mathcal{K}(1^\lambda)$, which on the security parameter λ, produces a pair of matching public and private keys $(\mathsf{pk}, \mathsf{sk})$; the encryption algorithm $\mathcal{E}_{\mathsf{pk}}(m; r)$ which outputs a ciphertext c corresponding to the plaintext $m \in \mathcal{M}$, using random coins $r \in \mathcal{R}$; and the decryption algorithm $\mathcal{D}_{\mathsf{sk}}(c)$ which outputs the plaintext m associated to the ciphertext c.

Security Notions *Semantic security* (*a.k.a. polynomial security* or *indistinguishability of encryptions* [11], denoted IND) is considered to be the standard notion: if the attacker has some *a priori* information about the plaintext, it should not learn more from the view of the ciphertext. More formally, this security notion requires the computational indistinguishability between two strings, chosen by the adversary, one of which has been encrypted. The adversary needs to decide which one has been actually encrypted with a probability significantly higher than one half: the advantage $\mathsf{Adv}_\pi^{\mathsf{ind}}(\mathcal{A})$, where the adversary $\mathcal{A}$ is seen as a 2-stage Turing machine $(\mathcal{A}_1, \mathcal{A}_2)$ whose running time is bounded by a polynomial function in λ, should be a negligible function in λ. Formally, with λ the security parameter:

$$\mathsf{Adv}_\pi^{\mathsf{ind}}(\mathcal{A}) = 2 \times \Pr\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^\lambda), (m_0, m_1, s) \leftarrow \mathcal{A}_1(\mathsf{pk}), \\ b \overset{R}{\leftarrow} \{0,1\}, c = \mathcal{E}_{\mathsf{pk}}(m_b) : \mathcal{A}_2(m_0, m_1, s, c) = b\end{array}\right] - 1.$$

On the other hand, an attacker can employ various kinds of attacks, depending on thee available information. Since we are considering asymmetric encryptions, the adversary can encrypt any plaintext of its choice with the public key, hence the basic *chosen-plaintext attack.* The strongest formalized attack can have a polynomial access to the decryption oracle (except the challenge ciphertext $c = \mathcal{E}_{\mathsf{pk}}(m_b)$), *adaptive chosen-ciphertext attacks* [16], denoted CCA or $\mathsf{CCA2}$ (by opposition to the earlier *non-adaptive chosen-ciphertext attacks* or *lunchtime attacks* or [15], denoted $\mathsf{CCA1}$, where the access to the decryption oracle is limited until the challenge is known.)

The strongest security notion for asymmetric encryptions is thus the *semantic security against adaptive chosen-ciphertext attacks* denoted $\mathsf{IND\text{-}CCA}$ or CCA security. In this paper, as we deal with this security notion, the adversary $\mathcal{A}$ is allowed to access the decryption oracle.
Denote $max_{\mathcal{A}}(\mathsf{Adv}_\pi^{\mathsf{ind}}(\mathcal{A}))$ by $\mathsf{Adv}_\pi^{\mathsf{ind-cca}}(\lambda)$. The CCA security of π requires that $\mathsf{Adv}_\pi^{\mathsf{ind-cca}}(\lambda)$ is a negligible function in λ.

3 Construction

In the following sections, one assumes having a group generator $\mathcal{G}$ which, on input 1^λ, returns (g, q) satisfying that g is a generator of a cyclic group $\mathcal{G}_q = \langle g \rangle$ of prime order q, and $2^{\lambda-1} < q < 2^\lambda$.

3.1 Damgård's ElGamal Encryption [5]

First of all, we recall Damgård's ElGamal Encryption.

Key Generation $\mathcal{K}(1^\lambda)$: $(g, q) \leftarrow \mathcal{G}(1^\lambda)$, random elements $x, y \in \mathbb{Z}_q$ are also chosen.
Secret Key: $\mathsf{sk} = (x, y)$
Public Key: $\mathsf{pk} = (g, c = g^x, d = g^y)$

Encryption: Given a message m, the encryption runs as follows. First, it chooses $r \overset{R}{\leftarrow} \mathbb{Z}_q$ and then computes:

$$u_1 = g^r, u_2 = c^r, e = d^r \cdot m$$

The outputted ciphertext is (u_1, u_2, e)

Decryption: Given a ciphertext (u_1, u_2, e), the decryption runs as follows. First, it tests if $u_2 = u_1^x$. If this condition does not hold, the decryption algorithm outputs "reject"; otherwise it computes:

$$K = u_1^y, \quad m = K^{-1} \cdot e$$

and outputs m.

3.2 Hybrid Damgård's ElGamal Encryption

We now propose a hybrid variant of the above Damgård's ElGamal Encryption. We just replace, in the encryption of Damgård's ElGamal scheme, the operation $e = d^r \cdot m$ by a symmetric encryption of m under the key $K = H(d^r)$, where H is randomly chosen from a universal familly of hash functions. The detailed description follows.
In our scheme, we make use of:

– a Data Encapsulation Mechanism (DEM). A DEM is a symmetric key encryption scheme, with encryption algorithm E and decryption algorithm D, such that for the key $K \in \mathcal{K}_D$ ($\mathcal{K}_D$ is key space of DEM whose size depends on λ) and the plaintext $m \in \{0,1\}^\star$, $e := E_K(m)$ is the encryption of m under K, and for key $K \in \mathcal{K}_D$ and the ciphertext $e \in \{0,1\}^\star$, $m := D_K(m)$ is the decryption of e under K.
For our purpose, we require that DEM is CCA secure, *i.e.* the advantage to distinguish $E_K(m_0)$ from $E_K(m_1)$ should be a negligible function in λ, (*i.e.* given the challenge ciphertext $E_K(m_b)$ for a random bit b, hard to guess b) for randomly chosen K and adversarially chosen m_0 and m_1 (where

m_0 and m_1 are of equal length but different), even though the adversary has access to the decryption oracle for its chosen ciphertext (except the challenge ciphertext). We denote this advantage by $\mathsf{Adv}_{\mathsf{dem}}^{\mathsf{ind-cca}}(\lambda)$.
Unlike the case of public-key, secure symmetric encryption schemes against chosen-ciphertext attacks can be easily built out of weaker primitives: all one needs is a secure symmetric encryption scheme against passive adversaries, and a secure message authentication code (MAC).
– a key derivation function H which maps an element $V \in \mathcal{G}_q$ to $H(V) \in \mathcal{K}_D$. We require that it's hard to distinguish $H(V)$ from K', where V and K' are both randomly chosen. We can thus use H as a function from a universal family of hash functions $\mathcal{H}_\lambda$and let $\mathcal{H} = \{\mathcal{H}_\lambda\}_{\lambda \in \mathbb{N}}$. In other words, we denote by $\mathsf{Adv}_{\mathcal{H}}^{\mathsf{ind}}(\lambda)$ the maximum advantage of all adversaries, whose running times are bounded by a polynomial function in λ, to distinguish $H(V)$ from K', where V and K' and $H \in \mathcal{H}_\lambda$ are randomly chosen, and assume that $\mathsf{Adv}_{\mathcal{H}}^{\mathsf{ind}}(\lambda)$ is a negligible function in λ. However, we will need some more requirements about H which will be described in Section 4.

We now describe our proposed scheme:

Key Generation $\mathcal{K}(1^\lambda)$: $(g, q) \leftarrow \mathcal{G}(1^\lambda)$, random elements $x, y \in \mathbb{Z}_q$ are also chosen. Next a hash function H is randomly chosen from a universal family of hash functions $\mathcal{H}_\lambda$.
Secret Key: $\mathsf{sk} = (x, y)$
Public Key: $\mathsf{pk} = (H, g, c = g^x, d = g^y)$
Encryption: Given a message m, the encryption runs as follows. First, it chooses $r \xleftarrow{R} \mathbb{Z}_q$ and then it computes:

$$u_1 = g^r, u_2 = c^r, K = H(d^r), e = E_K(m)$$

The ciphertext is (u_1, u_2, e)
Decryption: Given a ciphertext (u_1, u_2, e), the decryption runs as follows. First, it tests if $u_2 = u_1^x$. If this condition does not hold, the decryption algorithm outputs "reject"; otherwise it computes:

$$K = H(u_1^y), \quad m = D_K(e)$$

and outputs m.

3.3 Comparision with Kurosawa-Desmedt scheme

We briefly recall the Kurosawa-Desmedt scheme which makes use of:

– a CCA secure DEM.
– a key derivation function H which is a universal one-way hash function [9].
– a target collision resistance function $\mathsf{TCR} : \mathcal{G}_q \times \mathcal{G}_q \to \mathbb{Z}_q$: given $u_1 := g^{r_1}$ and $u_2 := g^{r_2}$, for random $r_1, r_2 \in \mathbb{Z}_q$, it is hard to find $(u_1', u_2') \in \mathcal{G}_q \times \mathcal{G}_q \setminus \{(u_1, u_2)\}$ such that $H(u_1', u_2') = H(u_1, u_2)$

We now describe their scheme: one uses a group generator $\mathcal{G}'$, which on input 1^λ, returns (g_1, g_2, q) where g_1, g_2 are two generators of the cyclic group $\langle g \rangle$ of order q, and $2^{\lambda-1} < q < 2^\lambda$.

Key Generation $\mathcal{K}(1^\lambda)$: $(g_1, g_2, q) \leftarrow \mathcal{G}(1^\lambda)$, random elements $x_1, x_2, y_1, y_2 \in \mathbb{Z}_q$ are also chosen.
  Secret Key: $\mathsf{sk} = (x_1, x_2, y_1, y_2)$
  Public Key: $\mathsf{pk} = (H, \mathsf{TCR}, g_1, g_2, c = g_1^{x_1} g_2^{x_2}, d = g_1^{y_1} g_2^{y_2})$

Encryption: Given a message m, the encryption runs as follows. First, it chooses $r \overset{R}{\leftarrow} \mathbb{Z}_q$ and then it computes:

$$u_1 = g_1^r, u_2 = g_2^r, \alpha = \mathsf{TCR}(u_1, u_2)$$

$$v = c^r d^{r\alpha}, K = H(v), e = E_K(m)$$

The ciphertext is (u_1, u_2, e)

Decryption: Given a ciphertext (u_1, u_2, e), the decryption runs as follows. First, it computes

$$\alpha = \mathsf{TCR}(u_1, u_2), v = u_1^{x_1 + y_1\alpha} u_2^{x_2 + y_2\alpha}, K = H(v)$$

and then $m = D_K(e)$ (m may be "reject").

Comparison. In terms of efficiency, our scheme has the following advantages over the Kurosawa-Desmedt one:

Key Size: The secret key contains two group elements, compared to four group elements in the Kurosawa-Desmedt scheme. Our scheme needs one group generator, compared to two group generators in Kurosawa-Desmedt scheme and does not need to use the TCR function, the public key is thus shorter.

Encryption Computation: We get rid of the TCR function. Moreover, in our scheme $v = d^r$ while in the Kurosawa-Desmedt scheme $v = c^r d^{r\alpha}$. We can thus save one exponentiation. A secondary advantage is that, in our scheme, the values u_1, u_2, d could be computed in parallel, while in the Kurosawa-Desmedt scheme, one should first compute u_1, u_2, then compute $\alpha = \mathsf{TCR}(u_1, u_2)$, and finally compute v.

Decryption Computation: We also get rid of the computation of the TCR function. Otherwise, both schemes need two exponentiations. However, in our scheme, as all exponentiations are with respect to the same base, the algorithm can be executed faster.

4 Assumptions used in the Security Analyses

4.1 Hashed Decisional Diffie-Hellman Assumption

We first recall the definition of the Hashed Diffie-Hellman (HDDH) problem needed for the sake of this work.

Assumption 1 (Hashed Decisional Diffie-Hellman Assumption)
Assume that there is no adversary that can effectively distinguish the two following distributions:

- $(g, q) \leftarrow \mathcal{G}(1^\lambda)$ *and a hash function* H *;*
- *the distribution* R_H *of random elements in* $\mathcal{G}_q^4$*:* $(g, g^a, g^b, H(Z))$*, for randomly distributed* $a, b \overset{R}{\leftarrow} \mathbb{Z}_q, Z \overset{R}{\leftarrow} \mathcal{G}_q$*;*
- *the distribution* D_H *of tuples elements in* $\mathcal{G}_q^4$*:* $(g, g^a, g^b, H(g^{ab}))$*, for randomly distributed* $a, b \overset{R}{\leftarrow} \mathbb{Z}_q$*.*

In other words, for all probabilistic algorithms $\mathcal{A}$ *whose running times are bounded by a polynomial function in* λ*, we define* $\mathsf{Adv}_{\mathcal{G}}^{\mathsf{hddh}}(\mathcal{A})$ *the advantage that* $\mathcal{A}$ *can distinguish the two above distributions. The* HDDH *assumption states that* $\mathsf{Adv}_{\mathcal{G}}^{\mathsf{hddh}}(\lambda) = max_{\mathcal{A}}(\mathsf{Adv}_{\mathcal{G}}^{\mathsf{hddh}}(\mathcal{A}))$ *is a negligible function in* λ*.*

For our scheme, we need to use the following variant of the HDDH assumption:

Assumption 2 (Modified Hashed Decisional Diffie-Hellman: MHDDH**)**
Assume that there is no adversary that can effectively distinguish the two following distributions:

- $(g, q) \leftarrow \mathcal{G}(1^\lambda)$ *and a hash function* H*;*
- *the distribution* R_{MH} *of random elements in* $\mathcal{G}_q^6$*:* $(g, g^a, g^b, g^c, g^{ac}, H(Z))$*, for randomly distributed* $a, b, c \overset{R}{\leftarrow} \mathbb{Z}_q, Z \overset{R}{\leftarrow} \mathcal{G}_q$*;*
- *the distribution* D_{MH} *of tuples elements in* $\mathcal{G}_q^6$*:* $(g, g^a, g^b, g^c, g^{ac}, H(g^{bc}))$*, for randomly distributed* $a, b, c \overset{R}{\leftarrow} \mathbb{Z}_q$*.*

In other words, for all probabilistic algorithms $\mathcal{A}$ *whose running times are bounded by a polynomial function in* λ*, we define* $\mathsf{Adv}_{\mathcal{G}}^{\mathsf{mhddh}}(\mathcal{A})$ *the advantage that* $\mathcal{A}$ *can distinguish the two above distributions. The* MHDDH *assumption states that* $\mathsf{Adv}_{\mathcal{G}}^{\mathsf{mhddh}}(\lambda) = max_{\mathcal{A}}(\mathsf{Adv}_{\mathcal{G}}^{\mathsf{mhddh}}(\mathcal{A}))$ *is a negligible function in* λ*.*

Proposition 3. *The* MHDDH *and the* HDDH *assumptions are equivalent:*

$$\mathsf{Adv}_{\mathcal{G}}^{\mathsf{hddh}}(\lambda) = \mathsf{Adv}_{\mathcal{G}}^{\mathsf{mhddh}}(\lambda)$$

Proof. The proof is quite trivial:

- Suppose that there is a MHDDH distinguisher for R_{MH} and D_{MH}, we construct an HDDH distinguisher for R_H and D_H as follows. Given an instance (g, g^b, g^c, Z), one randomly chooses $a \overset{R}{\leftarrow} \mathbb{Z}_q$, computes g^a, g^{ac}, then gives $(g, g^a, g^b, g^c, g^{ac}, Z)$ to MHDDH distinguisher and finally outputs what the MHDDH distinguisher returns. Thus:

$$\mathsf{Adv}_{\mathcal{G}}^{\mathsf{hddh}}(\lambda) \leq \mathsf{Adv}_{\mathcal{G}}^{\mathsf{mhddh}}(\lambda)$$

– Inversely, suppose that there is a HDDH distinguisher for R_H and D_H, we construct an MHDDH distinguisher for R_{MH} and D_{MH} as follows. Given an instance $(g, g^a, g^b, g^c, g^{ac}, U)$, one just gives (g, g^a, g^c, U) to HDDH distinguisher and outputs what the HDDH distinguisher returns. Thus:
$$\mathsf{Adv}_{\mathcal{G}}^{\mathsf{mhddh}}(\lambda) \leq \mathsf{Adv}_{\mathcal{G}}^{\mathsf{hddh}}(\lambda)$$
$\square$

We also need to introduce another assumption about the function H.

Assumption 4 (Extended Hashed Decisional Diffie-Hellman: EHDDH) *Assume that there is no adversary that can effectively distinguish the two following distributions R_{EH} and D_{EH} :*

– $(g, q) \leftarrow \mathcal{G}(1^\lambda)$ *and a hash function H;*
– *an element $U \in \mathcal{G}_q, U \neq 1$ and an element $v \in \mathbb{Z}_q^\star$ adversarially chosen;*
– *the distribution R_{EH} of random elements in $\mathcal{G}_q^4$: $(g, g^a, g^b, H(g^{ab}), H(Z))$, for randomly distributed $a, b \xleftarrow{R} \mathbb{Z}_q, Z \xleftarrow{R} \mathcal{G}_q$;*
– *the distribution D_{EH} of tuples elements in $\mathcal{G}_q^6$: $(g, g^a, g^b, H(g^{ab}), H(Ug^{abv}))$, for randomly distributed $a, b \xleftarrow{R} \mathbb{Z}_q$.*

In other words, for all probabilistic algorithms $\mathcal{A}$ whose running times are bounded by a polynomial function in λ, we define $\mathsf{Adv}_{\mathcal{G}}^{\mathsf{ehddh}}(\mathcal{A})$ the advantage that $\mathcal{A}$ can distinguish the two above distributions. The EHDDH assumption states that $\mathsf{Adv}_{\mathcal{G}}^{\mathsf{ehddh}}(\lambda) = max_{\mathcal{A}}(\mathsf{Adv}_{\mathcal{G}}^{\mathsf{ehddh}}(\mathcal{A}))$ is a negligible function in λ.

4.2 Diffie-Hellman Knowledge Assumption (DHK)

We now recall the definition of Diffie-Hellman Knowledge Assumptions.

For $A, B, C \in \mathcal{G}_q$, we say that (A, B, C) is a DH-triple if there exists $a, b \in \mathbb{Z}_q$ such that $A = g^a, B = g^b$ and $C = g^{ab}$. We say that (B, C) is a DH-pair relative to A if (A, B, C) is a DH-triple (throughout the text, when we say (A, B, C) is a DH-triple, we assume the base g to be fixed and known). We also write $C = DH(A, B)$. One way for an adversary $\mathcal{A}$ taking input g, A to output a DH-pair (B, C) relative to A is to pick (and thus "know") some $b \in \mathbb{Z}_q$, set $B = g^b$ and $C = A^b$, and output (B, C). Damgård [5] makes an assumption which, informally, implies that this is the "only" way that a polynomial-time adversary $\mathcal{A}$, given (g, A), can output a DH-pair (B, C) relative to A.

Bellare and Palacio [2] provide a formalization of this assumption that we refer to as the DHK (DHK stands for Diffie-Hellman Knowledge) assumption: the adversary $\mathcal{A}$, given (g, A), interacts with an extractor $\mathcal{A}^\star$, querying it adaptively, where $\mathcal{A}^\star$ is an algorithm that takes a pair of group elements and some state information (the state of $\mathcal{A}^\star$ is denoted by $St[\mathcal{A}^\star]$), and returns an exponent and a new state.

Experiment $\mathsf{Exp}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}^{\mathsf{dhk}}(\lambda)$
Below, $\mathcal{A}, \mathcal{A}^\star$ are polynomial-time probabilistic algorithms (whose running times are bounded by a polynomial function in λ).

– $(g, q) \leftarrow \mathcal{G}(1^\lambda)$; $a \overset{R}{\leftarrow} \mathbb{Z}_q, A = g^a$.
– Choose coins $R[\mathcal{A}], R[\mathcal{A}^\star]$ for $\mathcal{A}; \mathcal{A}^\star$, respectively ; $St[\mathcal{A}^\star] \leftarrow ((g, A), R[\mathcal{A}])$.
– Run $\mathcal{A}$ on input g, A and coins $R[\mathcal{A}]$ until it halts, replying to its oracle queries as follows:
  If $\mathcal{A}$ makes query (B, C) then:
  - $(b, St[\mathcal{A}^\star]) \leftarrow \mathcal{A}^\star((B, C), St[\mathcal{A}^\star]; R[\mathcal{A}^\star])$
  - If $C = B^a$ and $B \neq g^b$ then return 1,
    else return b to $\mathcal{A}$ as the reply.
– Return 0

Assumption 5 (DHK) *We define the* DHK*-advantage of $\mathcal{A}$ relative to $\mathcal{A}^\star$ as:*

$$\mathsf{Adv}^{\mathsf{dhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda) = \Pr[\mathsf{Exp}^{\mathsf{dhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda) = 1]$$

We say that $\mathcal{G}_q$ satisfies the DHK *assumption if for every polynomial-time* dhk*-adversary $\mathcal{A}$, there exists a polynomial-time* dhk*-extractor $\mathcal{A}^\star$ such that* $\mathsf{Adv}^{\mathsf{dhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)$ *is a negligible function in λ.*

We define $\mathsf{Adv}^{\mathsf{dhk}}_{\mathcal{G}}(\lambda) = max_{\mathcal{A}}(min_{\mathcal{A}^\star}(\mathsf{Adv}^{\mathsf{dhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)))$. The DHK assumption can be expressed as: $\mathsf{Adv}^{\mathsf{dhk}}_{\mathcal{G}}(\lambda)$ is a negligible function in λ.

4.3 Extended DHK Assumptions

We now consider an extension of the DHK assumption that we refer to as EDHK (EDHK stands for Extended Diffie-Hellman Knowledge). The adversary is now given not only (g, A) but also a DH-pair (B, C) relative to A. One way for an adversary $\mathcal{A}$ taking input (g, A), a pair (B, C) relative to A, to output a DH-pair (B', C') relative to A is to pick (and thus "know") some $x, y \in \mathbb{Z}_q$, and set $B' = B^x g^y$ and $C' = C^x A^y$.

Bellare and Palacio [1] introduced an assumption saying that this is the "only" way that a polynomial-time adversary $\mathcal{A}$, given g, A and a pair (B, C) relative to A, can output a DH-pair B', C' relative to A. Their formalization in [1] (called KEA3 assumption) is for non-randomized adversaries. For our purpose of dealing with probabilistic adversaries, we reuse the terms in [2] to formalize this assumption.

Considering an adversary $\mathcal{A}$, given (g, A) and a pair (B, C) relative to A, interacts with an extractor $\mathcal{A}^\star$, queries it adaptively, where $\mathcal{A}^\star$ is an algorithm that takes a tuple of group elements (g, A, B, C) and some state information (the state of $\mathcal{A}^\star$ is denoted by $St[\mathcal{A}^\star]$), and finally returns two exponents and a new state.

Experiment $\mathsf{Exp}^{\mathsf{edhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)$
Below, $\mathcal{A}, \mathcal{A}^\star$ are probabilistic algorithms whose running times are bounded by a polynomial function in λ.

– $(g, q) \leftarrow \mathcal{G}(1^\lambda)$; $a, b \overset{R}{\leftarrow} \mathbb{Z}_q, A = g^a, B = g^b, C = g^{ab}$.

– Choose coins $R[\mathcal{A}], R[\mathcal{A}^\star]$ for $\mathcal{A}; \mathcal{A}^\star$, respectively ; $St[\mathcal{A}^\star] \leftarrow ((g, A, B, C), R[\mathcal{A}])$.
– Run $\mathcal{A}$ on input g, A and coins $R[\mathcal{A}]$ until it halts, replying to its oracle queries as follows:
  If $\mathcal{A}$ makes query (B', C') then:
  - $(x||y, St[\mathcal{A}^\star]) \leftarrow \mathcal{A}^\star((B', C'), St[\mathcal{A}^\star]; R[\mathcal{A}^\star])$
  - If $(C = B^a$ and $C' = B'^a)$: if $(B' = B^x g^y$ and $C' = C^x A^y))$ then return $x||y$,
    else return 1, as the reply, to $\mathcal{A}$.
– Return 0

Assumption 6 (EDHK) *We define the* EDHK*-advantage of $\mathcal{A}$ relative to $\mathcal{A}^\star$ as:*

$$\mathsf{Adv}^{\mathsf{edhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda) = \Pr[\mathsf{Exp}^{\mathsf{edhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda) = 1]$$

We say that $\mathcal{G}_q$ satisfies the EDHK *assumption if for every polynomial-time* edhk*-adversary $\mathcal{A}$, there exists a polynomial-time* edhk*-extractor $\mathcal{A}^\star$ such that $\mathsf{Adv}^{\mathsf{edhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)$ is a negligible function in λ.*

We define $\mathsf{Adv}^{\mathsf{edhk}}_{\mathcal{G}}(\lambda) = max_{\mathcal{A}}(min_{\mathcal{A}^\star}(\mathsf{Adv}^{\mathsf{edhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)))$.The EDHK assumption can be expressed as: $\mathsf{Adv}^{\mathsf{edhk}}_{\mathcal{G}}(\lambda)$ is a negligible function in λ.

5 Security of the Hybrid Damgård's ElGamal Encryption

Theorem 7. *The Hybrid Damgård's ElGamal encryption is* CCA *secure assuming that:*

1. *the* HDDH *and* EHDDH *assumptions hold, and*
2. *the* DHK *and* EDHK *assumptions hold, and*
3. *the DEM is* CCA *secure.*

Proof. We use the Game hopping technique.

<u>Game $\mathbf{G}_0$:</u> The simulator runs the real IND-CCA attack game. The key generation algorithm in Damgård's ElGamal encryption scheme generates a secret key $\mathsf{sk} = (x, y)$ and a corresponding public key $\mathsf{pk} = (H, g, c = g^x, d = g^y)$. The simulator is given both the secret key $\mathsf{sk} = (x, y)$ and the public key pk and it generates random coins $R[\mathcal{A}]$ and $R[\mathcal{A}^\star]$. The simulator runs the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ on input pk and coins $R[\mathcal{A}]$. When $\mathcal{A}_1$ outputs a pair of messages (m_0, m_1), the simulator produces a challenge ciphertext by flipping a coin b and producing a ciphertext of m_b. This ciphertext $(u_1^\star, u_2^\star, e^\star)$ comes from a random string $r^\star \overset{R}{\leftarrow} \mathbb{Z}_q$:

$$u_1^\star = g^{r^\star}, u_2^\star = c^{r^\star}, K^\star = H(d^{r^\star}), e^\star = E_{K^\star}(m_b)$$

On input $(u_1^\star, u_2^\star, e^\star)$, $\mathcal{A}_2$ outputs bit b'. We note that the adversary can submit decryption queries in both stages, before and after receiving the challenge ciphertext.

We denote by S_0 the event $b' = b$ and use the same notation S_n in any game $\mathbf{G}_n$ below. Note that the adversary is given access to the decryption oracle $\mathcal{D}_{\mathsf{sk}}$ during both steps of the attack.

$$\Pr[\mathsf{S}_0] = \frac{1}{2} \times \left(\mathsf{Adv}^{\mathsf{ind-cca}}_{\pi}(\mathcal{A}) + 1\right).$$

<u>Game $\mathbf{G}_1$:</u> In this game, we simulate the decryption oracle without making use of the secret key. The input of the simulator is just the public key. In order to simulate the decryption queries, the simulator will use the DHK Extractor in the first stage (before receiving the challenge) and the EDHK Extractor in the second stage (after receiving the challenge $(u_1^\star, u_2^\star, e^\star)$). The state of the extractor $\mathcal{A}^\star$ in the first stage is set to be $St[\mathcal{A}^\star] \leftarrow ((g, c), R[\mathcal{A}])$ and the state of the extractor $\mathcal{A}^\star$ in the second stage will be set to be $St[\mathcal{A}^\star] \leftarrow ((g, c, u_1^\star, u_2^\star), R[\mathcal{A}])$. We now describe the simulation of the decryption oracle:

- Decryption queries in the first stage (the queries submitted by $\mathcal{A}_1$): whenever the adversary submits a query (u_1, u_2, e), the simulator runs the DHK Extractor $\mathcal{A}^\star((u_1, u_2), St[\mathcal{A}^\star], R[\mathcal{A}^\star])$. If the $\mathsf{Exp}^{\mathsf{dhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)$ outputs 0 or 1, the simulator rejects this ciphertext. If the $\mathsf{Exp}^{\mathsf{dhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)$ outputs r, the simulator computes $K = H(d^r)$ and outputs $m = D_K(m)$.
- Decryption queries in the second stage: whenever the adversary submits a query (u_1, u_2, e), the simulator uses EDHK Extractor $\mathcal{A}^\star((u_1, u_2), St[\mathcal{A}^\star], R[\mathcal{A}^\star])$. If the $\mathsf{Exp}^{\mathsf{edhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)$ outputs 0, the simulator rejects this ciphertext. If the $\mathsf{Exp}^{\mathsf{edhk}}_{\mathcal{G},\mathcal{A},\mathcal{A}^\star}(\lambda)$ outputs $(r_1||r_2)$:
  - $r_2 = 0$, the simulator simply computes $K = H(d^{r_1})$ and outputs $m = D_K(e^\star)$
  - if $r_2 \neq 0$: $u_1 = g^{r_1}(u_1^\star)^{r_2}$ and $u_2 = c^{r_1}(u_2^\star)^{r_2} = DH(c, u_1)$. The session key $K = H(DH(d, u_1)) = H(d^{r_1}(u_2^\star)^{r_2})$. The simulator decrypts e under a hashed random session key (hash of a random group element of $\mathcal{G}_q$) and returns the resulted plaintext to the adversary. Under EHDDH, it's easy to see that the adversary cannot distinguish $H(d^{r_1}(u_2^\star)^{r_2})$ from $H(R)$, where $R \overset{R}{\leftarrow} \mathcal{G}_q$.

Therefore, under the DHK, EDHK and EHDDH assumptions, the adversary cannot distinguish the two games $\mathbf{G}_1$ and $\mathbf{G}_0$:

$$|\Pr[\mathsf{S}_1] - \Pr[\mathsf{S}_0]| \leq \mathsf{Adv}^{\mathsf{dhk}}_{\mathcal{G}}(\lambda) + \mathsf{Adv}^{\mathsf{edhk}}_{\mathcal{G}}(\lambda) + n_q \mathsf{Adv}^{\mathsf{ehddh}}_{\mathcal{G}}(\lambda),$$

where n_q is the maximum number of decryption queries that the adversary could make.

<u>Game $\mathbf{G}_2$:</u> One now replaces the challenge $(u_1^\star = g^{r^\star}, u_2^\star = g^{r^\star x}, K^\star = H(g^{yr^\star}))$ by a random challenge $(u_1^\star, u_2^\star = g^{r^\star x}, K^\star = H(V^\star))$, where $V^\star \overset{R}{\leftarrow} \mathcal{G}$. The simulation of the decryption oracle is kept unchanged. Therefore, under the MHDDH assumption, *i.e.* it is hard to distinguish $(g, u_1^\star = g^{r^\star}, d =$

$g^y, c = g^x, u_2^\star = g^{r^\star x}, K^\star = H(g^{yr^\star}))$ from a random challenge $(g, u_1^\star, d = g^y, c = g^x, u_2^\star = g^{r^\star x}, K^\star = H(V^\star))$, the two games $\mathbf{G}_2$ and $\mathbf{G}_1$ are indistinguishable. Because of the equivalence between MHDDH and HDDH assumptions(Proposition 3), we have:

$$|\Pr[\mathsf{S}_2] - \Pr[\mathsf{S}_1]| \leq \mathsf{Adv}_{\mathcal{G}}^{\mathsf{hddh}}(\lambda),$$

<u>Game $\mathbf{G}_3$:</u> In this Game, we replace $K^\star$ by a random key in $\mathcal{K}_D$. In the previous game, the key $K^\star$ is chosen as $K^\star = H(V^\star)$ for a random $V^\star$, the distance between two games $\mathbf{G}_3$ and $\mathbf{G}_2$ is thus:

$$|\Pr[\mathsf{S}_3] - \Pr[\mathsf{S}_2]| \leq \mathsf{Adv}_{\mathcal{H}}^{\mathsf{ind}}(\lambda)$$

<u>Game $\mathbf{G}_4$:</u> Finally, we replace m_b in $e^\star = E_{K^\star}(m_b)$ by a random message $m^\star$. As the key $K^\star$ is randomly chosen, the distance between the two games $\mathbf{G}_4$ and $\mathbf{G}_3$ is:

$$|\Pr[\mathsf{S}_4] - \Pr[\mathsf{S}_3]| \leq \mathsf{Adv}_{\mathsf{dem}}^{\mathsf{ind-cca}}(\lambda)$$

In this game, the output of $\mathcal{A}_2$ follows a distribution that does not depend on b. Accordingly, $\Pr[\mathsf{S}_4] = 1/2$. $\square$

6 Hardness of the EDHK problem in the generic group model

The hardness of the DHK problem has been proven by Dent [6] in generic groups. In this section, we prove that the EDHK assumption also holds in generic groups. In the generic group model [19], elements of $\mathcal{G}_q$ appear to be encoded as unique random strings, so that no property other than equality can be directly tested by the adversary. An oracle is assumed to perform operations between group elements, *i.e.* the group action in the group $\mathcal{G}_q$. The encoding of the elements of $\mathcal{G}_q$ is modeled as an injective function $\xi : \mathbb{Z}_p \to \Sigma$, where $\Sigma \subset \{0,1\}^*$, which maps all $a \in \mathbb{Z}_p$ to the string representation $\xi(g^a)$ of $g^a \in G$.

Let us first recall a lemma [18, 19] that proofs in generic groups often rely on.

Lemma 8 ([18, 19]). *Let $F(x_1, x_2, ..., x_m)$ be a polynomial of total degree $d \geq 1$. Then the probability that $F(x_1, x_2, ..., x_m) = 0$ mod n for randomly chosen values $(x_1, x_2, ..., x_m)$ in $\mathbb{Z}_n$ is bounded above by d/p where p is the largest prime dividing n.*

We now show that the EDHK assumption holds in a generic group.

Theorem 9. *The* EDHK *assumption holds in a generic group.*

Proof. The extractor $\mathcal{A}^\star$ keeps track of the oracle queries of $\mathcal{A}$ as polynomials.

$\mathcal{A}^\star$ maintains a list of pair $L = \{(F_i, \xi_i) : i = 0, 1, ..., \tau - 1)\}$, where F_i are polynomials of degree ≤ 2 in $\mathbb{Z}_q[x, y, z_1, .., z_m]$. We set $F_0 = 1, F_1 = x, F_2 = y, F_3 = xy$ and $\tau = 4, m = 0$. The corresponding $\xi_0, \xi_1, \xi_2, \xi_3$ are set to be arbitrary distinct strings in $\{0, 1\}^*$. $\mathcal{A}^\star$ starts the game by providing $\mathcal{A}$ with the strings $\xi_0, \xi_1, \xi_2, \xi_3$. Each query of $\mathcal{A}$ is a group action.

$\mathcal{A}^\star$ answers group action queries as follows:

When $\mathcal{A}$ makes queries on strings that have not been in the list, those strings will be added to the list and assigned new variables: each time a new string appears, we denote it by ξ_τ and assign a new variable z_{m+1} to the element ξ_τ, we add $(F_\tau = z_m, \xi_\tau)$ to the list L and then increment τ and m by one.

We can now assume that $\mathcal{A}$ makes queries on strings in the list. Given a multiply/divide selection bit and two operands ξ_i, ξ_j with $0 \leq i, j \leq \tau - 1$, we compute $F_\tau = F_i + F_j \in \mathbb{Z}_q[x, y, z_1, .., z_m]$ or $F_\tau = F_i - F_j \in \mathbb{Z}_q[x, y, z_1, .., z_m]$ depending on whether a multiplication or a division is requested. If $F_\tau = F_\ell$ for some $\ell < \tau$, we set $\xi_\tau = \xi_\ell$; otherwise, we set ξ_τ to a string in $\{0, 1\}^*$ distinct from $\xi_0, \xi_1, ..., \xi_\tau$. We add (F_τ, ξ_τ) to L and give ξ_τ to $\mathcal{A}$, then increment τ by one.

$\mathcal{A}$ terminates and returns a pair ξ_i, ξ_j where $0 \leq i, j < \tau$. Let F_i, F_j be the corresponding polynomial in the list L. Note that if $\mathcal{A}$'s answer is correct then necessarily:

$$F_i(x, y, xy, z_1, z_2, ..., z_m) = xF_j(x, y, xy, z_1, z_2, ..., z_m)$$

Denote $F^\star(x, y, z_1, z_2, ..., z_\tau) = F_i(x, y, z_1, z_2, ..., z_\tau) - xF_j(x, y, xy, z_1, z_2, ..., z_\tau)$.

Case 1: $F^\star$ is identical to 0. From the above simulation, F_i, F_j should have the following form:

- $F_i(x, y, xy, z_1, z_2, ..., z_m) = c_{i_0} + \alpha_i x + \beta_i y + \gamma_i xy + c_{i_1} z_1 + ... + c_{i_m} z_m$
- $F_i(x, y, xy, z_1, z_2, ..., z_m) = c_{j_0} + \alpha_j x + \beta_j y + \gamma_j xy + c_{j_1} z_1 + ... + c_{j_m} z_m$

Therefore: $F^\star(x, y, z_1, z_2, ..., z_\tau) = c_{i_0} + (\alpha_i - c_{j_0})x + (\gamma_i - \beta_j)xy + \beta_i y - \alpha_j x^2 - \gamma_j xz + c_{i_1} z_1 + ... + c_{i_m} z_m - c_{j_1} xz_1 - ... - c_{j_m} xz_m - \gamma_j x^2 y$. We deduce thus:

- $\beta_i = \gamma_j = c_{i_1} = ... = c_{i_m} = c_{j_1} = ... = c_{j_m} = 0$
- $\alpha_i = c_{j_0} (= r_1)$
- $\gamma_i = \beta_j (= r_2)$

$F_i = r_1 x + r_2 xy, F_j = r_1 + r_2 y$. As F_i, F_j have this form, $\mathcal{A}^\star$ could easily extract and outputs $r_1 || r_2$.

Case 2: $F^\star$ is not identical to 0. In this case, $F^\star(x, y, z_1, z_2, ..., z_\tau) = 0$ is a non-trivial equation. At this point, $\mathcal{A}^\star$ chooses randomly $x^\star, y^\star, z_1^\star, z_2^\star, ..., z_m^\star$. The simulation provided by B is perfect unless the instantiation $x \leftarrow x^\star, y \leftarrow y^\star, z_1 \leftarrow z_1^\star, ..., z_m \leftarrow z_m^\star$ creates an equality relation between the simulated group elements that was not revealed to $\mathcal{A}$. The success probability of $\mathcal{A}$ is thus bounded by the probability that any of the following holds:

- $F_i(x^\star, y^\star, z_1^\star, ..., z_m^\star) = F_j(x^\star, y^\star, z_1^\star, ..., z_m^\star)$, for some $0 \leq i, j < \tau$. From Lemma 8, this occurs with a probability bounded by $\tau^2 2/q$.

– $F^\star(x^\star, y^\star, z_1^\star, ..., z_m^\star) = 0$. From Lemma 8, this occurs with a probability bounded by $1/q$.

□

7 Conclusion

We propose a hybrid Damgård's ElGamal encryption that is CCA secure. The proposed scheme is very efficient but its security should be based on an extension of the DHK assumption, which is quite strong. There are however some reasons that could convince us about the usefulness of this scheme. Firstly, we proved that the Extended DHK assumption holds in generic groups. Secondly, Gjøsteen[10] proposed a new technique of security proof for Damgård's ElGamal scheme (against non-adaptive chosen ciphertext) in which the DHK assumption can be replaced by an assumption of hardness of a new problem, the gap subgroup membership problem, which is somewhat similar to conventional problems such as the Gap Diffie-Hellman problem. Therefore, it could be possible that the Gjøsteen's technique (or another new technique) would help to replace the EDHK assumption by a more conventional one[3]. Finally, note that the HDDH assumption is generally weaker than DDH assumption and might hold even in groups where DDH problem is easy [8] and that EHDDH, EDHK assumption, though strong, might also hold in groups where the DDH problem is easy (it seems easy to prove the hardness of these assumptions in generic groups with pairings, for example). Therefore, it might happen that our proposed scheme is still secure in some non-DDH groups.

Acknowledgments

We would like to thank Eike Klitz for helpful discussions.

References

1. M. Bellare and A. Palacio. The knowledge-of-exponent assumptions and 3-round zero-knowledge protocols. In *Advances in Cryptology – Proceedings of CRYPTO '04*, volume LNCS 3152, pages 273–289, Berlin, 2004. Springer-Verlag.
2. M. Bellare and A. Palacio. Towards plaintext-aware public-key encryption without random oracles. In *Advances in Cryptology – Proceedings of ASIACRYPT '04*, volume LNCS 3329, pages 48–62, Berlin, 2004. Springer-Verlag.
3. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25, 1998. Springer-Verlag, Berlin, Germany.

[3] An independent work recently posted on ePrint [12] shows indeed that hybrid Damgård's ElGamal encryption can be proved under standard assumptions.

4. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.
5. I. Damgård. Towards Practical Public Key Systems Secure Against Chosen Ciphertext Attacks. In *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456, Santa Barbara, CA, USA, Aug. 11–15, 1992. Springer-Verlag, Berlin, Germany.
6. A. W. Dent. The Hardness of the DHK Problem in the Generic Group Model. Cryptology ePrint Archive, Report 2006/156, 2006. `http://eprint.iacr.org/`.
7. T. ElGamal. A Public-key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Trans. on Information Theory*, IT-31(4):469–472, 1985.
8. R. Gennaro, H. Krawczyk, and T. Rabin. Secure Hashed Diffie-Hellman over Non-DDH Groups. In *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag, Berlin, Germany.
9. R. Gennaro and V. Shoup. A note on an encryption scheme of Kurosawa and Desmedt. Cryptology ePrint Archive, Report 2004/194, 2004. `http://eprint.iacr.org/`.
10. K. Gjøsteen. A new security proof for damgård's ElGamal. In *Topics in Cryptology – CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 150–158, 2006. Springer-Verlag, Berlin, Germany.
11. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.
12. E. Kiltz, K. Pietrzak, M. Stam, and M. Yung. Randomness extraction: A new paradigm for hybrid encryption. Cryptology ePrint Archive, Report 2008/304, 2008. `http://eprint.iacr.org/`.
13. K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442, 2004. Springer-Verlag, Berlin, Germany.
14. H. Lipmaa. On CCA1-Security of Elgamal And Damgård Cryptosystems. Cryptology ePrint Archive, Report 2008/234, 2008. `http://eprint.iacr.org/`.
15. M. Naor and M. Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attack. In *22nd Annual ACM Symposium on Theory of Computing*, pages 427–437, 1990. ACM Press.
16. C. Rackoff and D. R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444, 1992. Springer-Verlag, Berlin, Germany.
17. R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978.
18. J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.
19. V. Shoup. Lower Bounds for Discrete Logarithms and Related Problems. In *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266, 1997. Springer-Verlag, Berlin, Germany.
20. V. Shoup. Using Hash Functions as a Hedge against Chosen Ciphertext Attack. In *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 275–288, 2000. Springer-Verlag, Berlin, Germany.
21. J. Wu and D. Stinson. On The Security of The ElGamal Encryption Scheme and Damgards Variant. Cryptology ePrint Archive, Report 2008/200, 2008. `http://eprint.iacr.org/`.