A Robust and Plaintext-Aware Variant of Signed ElGamal Encryption*

Yannick Seurin and Joana Treger

ANSSI, Paris, France
yannick.seurin@m4x.org,joana.marim@ssi.gouv.fr

Revised, 24 February 2013

Abstract. Adding a Schnorr signature to ElGamal encryption is a popular proposal aiming at thwarting chosen-ciphertext attacks by rendering the scheme *plaintext-aware*. However, there is no known security proof for the resulting scheme, at least not in a weaker model than the one obtained by combining the Random Oracle Model (ROM) and the Generic Group Model (Schnorr and Jakobsson, ASIACRYPT 2000). In this paper, we propose a very simple modification to Schnorr-Signed ElGamal encryption such that the resulting scheme is semantically secure under adaptive chosen-ciphertext attacks (IND-CCA2-secure) in the ROM under the Decisional Diffie-Hellman assumption. In fact, we even prove that our new scheme is plaintext-aware in the ROM as defined by Bellare *et al.* (CRYPTO '98). Interestingly, we also observe that Schnorr-Signed ElGamal is *not* plaintext-aware (again, for the definition of Bellare *et al.*) under the Computational Diffie-Hellman assumption. We show that our new scheme additionally achieves anonymity as well as robustness, a notion formalized by Abdalla *et al.* (TCC 2010) which captures the fact that it is hard to create a ciphertext that is valid under two different public keys. Finally, we study the hybrid variant of our new proposal, and show that it is IND-CCA2-secure in the ROM under the Computational Diffie-Hellman assumption when used with a symmetric encryption scheme satisfying the weakest security notion, namely ciphertext indistinguishability under one-time attacks (IND-OT-security).

Keywords: ElGamal encryption, Schnorr signature, chosen-ciphertext attacks, plaintext-aware encryption, robust encryption, hybrid encryption

1 Introduction

ElGamal and variants. The ElGamal encryption scheme [ElG85] is one of the oldest discrete-log based public key encryption scheme. It works as follows. Given a cyclic group $\mathbb{G}$ of prime order p and a generator G, a secret/public key pair is a pair $(x, X = G^x)$, where x is randomly drawn in $\mathbb{Z}_p^*$. To encrypt a message $M \in \mathbb{G}$, one draws a random integer $r \in \mathbb{Z}_p^*$, and computes $R = G^r$ and $Y = MX^r$. The ciphertext is (Y, R), and can be decrypted by computing Y/R^x.

ElGamal encryption is one-way under the Computational Diffie-Hellman (CDH) assumption, and its semantic security against chosen-plaintext attacks (IND-CPA-security) is equivalent to the Decisional Diffie-Hellman (DDH) assumption [TY98]. However, it is not secure against adaptive[1] chosen-ciphertext attacks since it is malleable: given a ciphertext (Y, R) corresponding to plaintext M, one can easily generate a ciphertext corresponding to MG^z as (YG^z, R). This implies that the scheme cannot be IND-CCA2-secure.

There have been mainly two approaches aiming at enhancing the security of ElGamal encryption to resist adaptive chosen-ciphertext attacks in the Random Oracle Model (ROM). The first one is to use the so-called *hybrid* (or *hashed*) variant of the scheme, where X^r is hashed through a

* An abridged version appears at CT-RSA 2013. This is the full version.
[1] Regarding non-adaptive chosen-ciphertext attacks, *i.e.* IND-CCA1 security of ElGamal encryption, see [Lip10].

random oracle $\boldsymbol{H_K}$ to give a secret key K subsequently used in a symmetric encryption scheme. The resulting scheme is often referred to as DHIES [ABR01, CS03, KM04]. To establish the IND-CCA2-security of hybrid ElGamal encryption, one has to rely on the IND-CCA2-security of the symmetric encryption scheme[2] and on the non-standard Strong Diffie-Hellman (SDH) assumption, which states that the CDH problem remains hard even given access to an oracle solving the DDH problem.[3] Recently, Cash *et al.* [CKS08] removed the need to rely on the SDH assumption by proposing the Twin ElGamal encryption scheme, a variant of hybrid ElGamal which is IND-CCA2-secure under the CDH assumption.

The second approach is to add a non-interactive proof of knowledge of the random integer r used to encrypt the plaintext, with the hope to make the scheme *plaintext-aware*. The notion of plaintext awareness was introduced in [BR94] and captures the intuitive idea that it should be impossible for an attacker to create a valid ciphertext without knowing the corresponding plaintext, thereby rendering the decryption oracle available to an IND-CCA2 adversary useless. The most natural idea is to add a Schnorr signature [Sch91] to the ciphertext, as was proposed in [Jak98, TY98]: to encrypt a message $M \in \mathbb{G}$, in addition to the usual ElGamal ciphertext $R = G^r$, $Y = MX^r$, a Schnorr signature (with secret key r) is computed using a random oracle $\boldsymbol{H_c}$ by drawing a random integer $a \in \mathbb{Z}_p^*$, and setting $A = G^a$, $c = \boldsymbol{H_c}(Y, R, A)$, and $s = a + cr \mod p$. The ciphertext is then (Y, R, c, s). To decrypt with the secret key x, first check whether $c = \boldsymbol{H_c}(Y, R, G^s R^{-c})$, then return $Y/R^x = M$ (or $\perp$ if the check failed). We call this scheme Schnorr-Signed ElGamal (SS-EG for short) encryption.[4]

Intuitively, since the Schnorr proof of knowledge is extractable in the ROM [PS96, PS00], the security reduction should be able to extract the integer r used to form a ciphertext (Y, R, c, s) and answer decryption queries issued by an IND-CCA2 adversary without knowing the secret key x as Y/X^r. However, no one knows how to turn this intuition into a formal security proof. As clearly explained by Shoup and Gennaro [SG02, Sec. 7.2], the problem is that Schnorr signatures are not known to be *online* (*a.k.a. straight-line*) extractable [Pas03, Fis05]: the extractor needs to rewind the prover (here, the adversary) in order to extract r. This causes problems if the adversary orders its random oracle and decryption queries in a certain way (*e.g.*, if it asks n random oracle queries $c_i = \boldsymbol{H_c}(Y_i, R_i, A_i)$ corresponding to the Schnorr signature for the encryption of n messages M_i, and then asks the decryption queries for the messages in reverse order, that is for $M_n, \ldots, M_1$). Tsiounis and Yung [TY98] gave a proof that SS-EG encryption is IND-CCA2-secure using a strong assumption on Schnorr signatures (which basically amounts to saying that they are online extractable). Schnorr and Jakobsson [SJ00] showed that SS-EG encryption is IND-CCA2-secure by combining the ROM and the Generic Group Model (GGM). However, as noted by Fischlin [Fis00], combining these two idealized models may be even more problematic than considering each one independently.

Shoup and Gennaro [SG02] proposed two variants of SS-EG encryption named TDH1 and TDH2, whose hybrid variants are IND-CCA2-secure in the ROM under respectively the CDH and the DDH assumption (regarding non-hybrid variants, we show that they are IND-CCA2-secure in Appendix C). They were primarily concerned with the context of threshold encryption, where it is necessary that

[2] In fact, the weaker notion of indistinguishability under one-time chosen-ciphertext attacks (IND-OTCCA) is sufficient [CS03, HHK10].

[3] The SDH assumption is slightly weaker than the Gap Diffie-Hellman (GDH) assumption [OP01]: the first element of the triplets submitted to the DDH oracle is fixed for the SDH problem, whereas it can be freely chosen for the GDH problem.

[4] A variant of the scheme is obtained by using (Y, R, A, s) for the ciphertext, *i.e* appending the traditional variant of the Fiat-Shamir signature consisting of the challenge and the response. Since integers mod p are usually more compact than group elements, the optimized variant of the signature consisting of the challenge and the response (c, s) is often preferred. This has no impact on security for SS-EG, but we warn that this will not be the case for the new scheme presented in Section 3.

decryption servers be able to publicly verify (*i.e.* without knowing the secret key) the validity of a ciphertext before starting the decryption process. This constraint leads to schemes which, though efficient, are more complex than SS-EG. It is therefore a natural question whether removing the necessity to be able to publicly verify the validity of a ciphertext could yield more efficient provably IND-CCA2-secure variants of SS-EG encryption.

Robustness. Robustness for an encryption scheme informally means that it is hard to create a ciphertext that decrypts to a valid plaintext under two different secret keys.[5] This notion was only recently formalized by Abdalla *et al.* [ABN10], and further studied by Mohassel [Moh10]. Robustness can always be achieved by appending the public key of the intended receiver to the ciphertext (and checking it on decryption). However, the notion becomes really interesting when coupled with anonymity (*a.k.a.* key privacy) [BBDP01], which requires that a ciphertext does not reveal the public key under which it was created—and then the previous solution for robustness becomes inadequate. Anonymity and robustness not only help make encryption more resistant to misuse, but also are inherently important properties for some applications of public key encryption such as encryption with keyword search [ABC$^+$08] or auction protocols [Sak00].

Basic ElGamal encryption can be shown anonymous under CPA-attacks assuming the DDH problem is hard [BBDP01], however it is clearly not robust since any ciphertext is valid. This is also true for Schnorr-Signed ElGamal[6] since the validity check does not depend on the secret nor the public key (anonymity and robustness for TDH1/2 are more subtle, but neither of them achieves both anonymity and robustness, see Appendix C). It is claimed in [ABN10] that a very minimal modification to DHIES [ABR01] (which is anonymous under CCA2-attacks) renders the scheme robust. Though no formal proof is available, such a proof seems to require non-standard assumptions on the MAC scheme used.[7] We also note that all variants of ElGamal encryption with compact ciphertexts where the decryption algorithm never rejects [KM04, Boy07, AKO09] cannot be robust (independently of whether they are anonymous or not). Hence, it seems to be an open problem to propose a simple variant of ElGamal encryption achieving both anonymity under CCA2-attacks and robustness without additional non-standard assumption.

Contributions of this work. We propose a very simple variant of Schnorr-Signed ElGamal encryption, and prove that it is IND-CCA2-secure in the ROM under the DDH assumption by showing that it is IND-CPA-secure and plaintext-aware. Additionally, we show that the resulting encryption scheme is not only anonymous under CCA2 attacks assuming the DDH problem is hard, but also strongly robust (assuming the additional check at decryption that the randomness used to encrypt was not 0). The proof of robustness only requires that the hash function used to instantiate the random oracle be collision-resistant.

The modification only affects the way the random challenge c for the signature is generated; in particular the key size remains unchanged compared with Schnorr-Signed ElGamal, whereas the ciphertext size is the same as for the unoptimized variant of Schnorr-Signed ElGamal (the optimized variant of the signature cannot be used as for SS-EG because in that case this leads to an insecure scheme, see Remark 1 in Section 3). We name our new scheme Chaum-Pedersen-Signed ElGamal (CPS-EG for short) encryption since it uses a proof of *equality* of discrete logarithms (introduced by Chaum and Pedersen [CP92]) rather than a proof of knowledge of discrete logarithm. This idea

[5] This is strong robustness. Weak robustness means that it is impossible to produce a plaintext that, once encrypted under some public key, decrypts to some valid plaintext under the secret key corresponding to a second, different public key.

[6] Schnorr-Signed ElGamal can easily be seen to achieve anonymity under CPA-attacks. We expect however that proving anonymity under CCA2-attacks will run into the same problems as a proof of IND-CCA2-security.

[7] Personal communication with the authors of [ABN10].

already proved fruitful for signature schemes [KW03, GJKW07]. In fact, our scheme can be seen as an optimized variant of TDH2 [SG02]. A comparison of CPS-EG with existing related schemes can be found in Table 1.

We prove our scheme to be in a sense *minimal* relatively to plaintext awareness: removing any input to the random oracle when computing the challenge for the signature makes the scheme provably (under some computational assumption) *not* plaintext-aware.[8] In particular, we show that Schnorr-Signed ElGamal encryption is *not* plaintext-aware as defined in [BDPR98] under the CDH assumption. We think that this observation might help explain why progress has remained elusive regarding IND-CCA2-security of this scheme.

We also analyze the hybrid version of our scheme, named HCPS-EG, and show that it is IND-CCA2-secure in the ROM under the CDH assumption. As the non-hybrid variant, the scheme is anonymous under CCA2-attacks (assuming the CDH problem is hard) and strongly robust (assuming a collision-resistant hash function). An interesting feature of this scheme is that it achieves IND-CCA2-security using the weakest form of security for the data encapsulation mechanism (DEM), namely ciphertext indistinguishability under one-time attacks (IND-OT-security) [CS03, HHK10]. In more concrete terms, the HCPS-EG scheme can be safely used with AES in counter mode. This is a property shared with other ElGamal variants, notably [Boy07, AKO09]. However for these two schemes, the decryption algorithm never rejects (they aim at compact ciphertexts), and hence they cannot be robust. A summary of results regarding HCPS-EG as well as a comparison with related schemes can be found in Table 1. Fully detailed analysis is deferred to Appendix B.

Related work. Abe [Abe02] studied generic ways to combine a hybrid encryption scheme and a signature scheme (which in particular applies to Hashed ElGamal and Schnorr signatures) to obtain an IND-CCA2 encryption scheme in the ROM where validity of a ciphertext can be publicly checked. Besides, a lot of efforts were devoted to variants of ElGamal encryption provably IND-CCA2-secure in the standard model. This started with the "double-base" variant by Damgård [Dam91], which is IND-CCA1-secure under the non-standard Diffie Hellman Knowledge (DHK) assumption [BP04]. Later, Cramer and Shoup [CS98] proposed their famous cryptosystem provably IND-CCA2-secure under the DDH assumption, and formally introduced the notion of hybrid encryption [CS03]. Notable subsequent work includes [KD04, KPSY09].

Organization. In Section 2, we recall the necessary background on ElGamal encryption and plaintext awareness. We describe our new scheme and prove that it is plaintext-aware, hence IND-CCA2-secure in Section 3. Then, we show in Section 4 that our scheme is in a sense minimal if one wants to obtain plaintext awareness. In Section 5, we study anonymity and robustness of our new scheme.

2 Preliminaries

2.1 Basic Definitions

The security parameter will be denoted k. When S is a non-empty finite set, we write $s \leftarrow_{\$} S$ to mean that a value is sampled uniformly at random from S and assigned to s. By $z \leftarrow \mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2,\cdots}(x, y, \ldots)$ we denote the operation of running the (possibly probabilistic) algorithm $\mathcal{A}$ on inputs $x, y, \ldots$ with access to oracles $\mathcal{O}_1, \mathcal{O}_2, \ldots$ (possibly none), and letting z be the output. PPT will stand for probabilistic polynomial-time.

A (prime-order) group generator `GpGen` is a PPT algorithm that takes a security parameter 1^k and outputs a triplet $(\mathbb{G}, p, G)$ where $\mathbb{G}$ is a group[9] of prime order $p \in [2^{k-1}, 2^k[$ and G is a generator

[8] We stress that we refer here to the strong notion of plaintext awareness in the ROM as defined in [BDPR98]. Our results seem unlikely to be extensible to weaker notions such as the PA2-RO notion suggested in [BP04].

[9] More precisely it outputs the description of the group, but we confound it here for simplicity.

Table 1. Comparison of variants of basic (non-hybrid) ElGamal encryption. The secret, resp. public key size is in number of mod p integers, resp. group elements. The next two columns give the number of exponentiations per encryption and decryption. For encryption, we separate off-line/online exponentiations. We count $G^s R^{-c}$ as one single exponentiation even though it is slightly more expensive. The ciphertext size is given in number of group elements plus mod p integers. The IND column gives the assumption needed to prove IND-CCA2-security (all schemes except ElGamal use the ROM). The ANON+SROB column indicates whether the scheme achieves ANON-CCA2-security and strong robustness, as well as the assumptions needed. DDH stands for Decisional Diffie-Hellman, GGM for Generic Group Model, and CRHF for Collision-Resistant Hash Function. Regarding the security properties of TDH1/2, see Appendix C.

scheme	`sk`/`pk` size	exp./enc.	exp./dec.	cip. size	IND	ANON+SROB	Refs
ElGamal	1	2/0	1	$2G$	DDH (CPA only)	No	[ElG85, TY98]
SS-EG	1	3/0	2	$2G + 2p$	GGM	No	[SJ00]
TDH1	1	3/2	3	$3G + 2p$	DDH	No	[SG02]
TDH2	1/2	5/0	3	$3G + 2p$	DDH	No	[SG02]
CPS-EG	1	4/0	4	$3G + p$	DDH	Yes (DDH+CRHF)	Sec. 3 & 5

Table 2. Comparison of variants of hashed ElGamal encryption. The first three columns are as for Table 1. The ciphertext expansion is given in number of group elements, plus the size $|\tau|$ of a MAC in case an authenticated DEM is needed. The DEM column indicates with which type of Data Encapsulation Mechanism the scheme is assumed to be used (see App. A for definitions of AE-OT and IND-OT). SPRP means that the DEM must be a (variable-input length) strong pseudorandom permutation. The IND and ANON+SROB columns are as for Table 1. For (Twin-)DHIES, the star indicates that a non-standard assumption is needed for the MAC to achieve strong robustness. CDH stands for Computational Diffie-Hellman, SDH for Strong DH, and GDH for Gap DH.

scheme	`sk`/`pk` size	exp./enc.	exp./dec.	cip. exp.	DEM	IND	ANON+SROB	Refs
DHIES	1	2	1	$G + \|\tau\|$	AE-OT	SDH	Yes*	[ABR01]
Twin-DHIES	2	3	2	$G + \|\tau\|$	AE-OT	CDH	Yes*	[CKS08]
KM	1	2	1	G	SPRP	SDH	No	[KM04]
Twin-KM	2	3	2	G	SPRP	CDH	No	[CKS08]
AKO	1	2	1	G	IND-OT	SDH	No	[AKO09]
Twin AKO	2	3	2	G	IND-OT	CDH	No	[AKO09]
Boyen	1	3	2	G	IND-OT	GDH	No	[Boy07]
HCPS-EG	1	4	3	$G + 2p$	IND-OT	CDH	Yes (CDH+CRHF)	App. B

of $\mathbb{G}$. In all the following, we will assume that all algorithms are given $(\mathbb{G}, p, G)$ (*e.g.*, as part of a public key) and will not denote it explicitly. We denote $1_{\mathbb{G}}$ the identity element of $\mathbb{G}$.

We say that the Computational Diffie-Hellman (CDH) problem is hard relatively to $\mathtt{GpGen}$ if the following advantage:

$$\mathbf{Adv}^{\mathrm{cdh}}_{\mathtt{GpGen},\mathcal{A}}(k) = \Pr\left[(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k), x, y \leftarrow_{\$} \mathbb{Z}_p, Z \leftarrow \mathcal{A}(G^x, G^y) : Z = G^{xy}\right]$$

is negligible for any PPT adversary $\mathcal{A}$.

We say that the Decisional Diffie-Hellman (DDH) problem is hard relatively to $\mathtt{GpGen}$ if the following advantage:

$$\mathbf{Adv}^{\mathrm{ddh}}_{\mathtt{GpGen},\mathcal{A}}(k) = |\Pr\left[(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k), x, y \leftarrow_{\$} \mathbb{Z}_p : 1 \leftarrow \mathcal{A}(G^x, G^y, G^{xy})\right] - \Pr\left[(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k), x, y, z \leftarrow_{\$} \mathbb{Z}_p : 1 \leftarrow \mathcal{A}(G^x, G^y, G^z)\right]|$$

is negligible for any PPT adversary $\mathcal{A}$.

Random oracles are used for three distinct goals in the various schemes considered in this paper, and will be denoted as follows: $\boldsymbol{H_c}$ will denote the random oracle used to generate the challenge for the signature, $\boldsymbol{H_K}$ will denote the random oracle used to derive the key for the DEM in hybrid schemes, and $\boldsymbol{H_G}$ will denote a random oracle mapping to $\mathbb{G}$ (only used in TDH1). We will use $\boldsymbol{H}$ as a shortcut for the set of random oracles accessed by a scheme.

Definition 1 (Encryption scheme). *A public key encryption scheme* $\mathtt{PKE}$ *is a triplet of polynomial-time algorithms* $(\mathtt{PKE.Kg}, \mathtt{PKE.Enc}, \mathtt{PKE.Dec})$ *where:*

- $\mathtt{PKE.Kg}$*, the (probabilistic)* key generation algorithm*, takes a security parameter* 1^k *and returns a secret/public key pair* $(\mathtt{sk}, \mathtt{pk})$*.*
- $\mathtt{PKE.Enc}$*, the (probabilistic)* encryption algorithm*, takes a public key* $\mathtt{pk}$ *and a message* M *and returns a ciphertext* ψ*.*
- $\mathtt{PKE.Dec}$*, the (deterministic)* decryption algorithm*, takes a secret key* $\mathtt{sk}$ *and a ciphertext* ψ *and returns either a message* M *or the special symbol* $\perp$ *that indicates that the ciphertext is invalid.*

We assume that a public key $\mathtt{pk}$ defines a message space $\mathtt{MsgSp}(\mathtt{pk})$ and for consistency we impose that for any k:

$$\Pr\left[(\mathtt{sk}, \mathtt{pk}) \leftarrow \mathtt{PKE.Kg}(1^k), M \leftarrow_{\$} \mathtt{MsgSp}(\mathtt{pk}), \psi \leftarrow \mathtt{PKE.Enc}(\mathtt{pk}, M) : \mathtt{PKE.Dec}(\mathtt{sk}, \psi) = M\right] = 1 \ .$$

We recall the usual security definitions for PKE schemes in Appendix A.

The ElGamal PKE encryption scheme is formally defined in Figure 1 (recall that the group parameters $(\mathbb{G}, p, G)$ are implicitly included in $\mathtt{pk}$, and here $\mathtt{MsgSp}(\mathtt{pk}) = \mathbb{G}$).

The following classical security result regarding ElGamal encryption is due to Tsiounis and Yung.

Theorem 1 ([TY98]). *The ElGamal encryption scheme is IND-CPA-secure if and only if the DDH problem is hard relatively to* $\mathtt{GpGen}$*.*

Proof. We recall how to prove that hardness of DDH implies that ElGamal is IND-CPA-secure. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an IND-CPA adversary against ElGamal. We build a reduction $\mathcal{R}$ solving the DDH problem. Let $(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k)$ and $(X = G^x, R = G^r, R')$ be the input to $\mathcal{R}$. $\mathcal{R}$ runs $\mathcal{A}_1$ with input $\mathtt{pk} = X$, which outputs two plaintexts M_0, M_1. $\mathcal{R}$ draws $b \leftarrow_{\$} \{0, 1\}$, and gives $\psi^* = (M_b R', R)$ as the challenge ciphertext to $\mathcal{A}_2$. If $\mathcal{A}_2$ outputs $b' = b$, then $\mathcal{R}$ outputs 1, otherwise it outputs 0.

ElGamal PKE scheme

$\texttt{PKE.Kg}(1^k)$	$\texttt{PKE.Enc}(\texttt{pk} = X, M)$	$\texttt{PKE.Dec}(\texttt{sk} = x, \psi)$
$(\mathbb{G}, p, G) \leftarrow \texttt{GpGen}(1^k)$	$r \leftarrow_\$ \mathbb{Z}_p^*$	Parse ψ as (Y, R)
$x \leftarrow_\$ \mathbb{Z}_p^*$; $X := G^x$	$R := G^r$; $R' := X^r$	$R' := R^x$
$\texttt{sk} := x$; $\texttt{pk} := X$	$Y := MR'$	Return $M := Y/R'$
Return $(\texttt{sk}, \texttt{pk})$	Return $\psi := (Y, R)$	

Fig. 1. The ElGamal encryption scheme.

Clearly, when (X, R, R') is a DDH tuple, the IND-CPA security experiment is perfectly simulated by $\mathcal{R}$, so that $b' = b$ with probability greater than $1/2 + \mathbf{Adv}^{\text{ind-cpa}}_{\texttt{PKE},\mathcal{A}}(k)$. When (X, R, R') is a random triplet then the view of $\mathcal{A}$ is independent of b and $b' = b$ with probability exactly $1/2$. Hence $\mathbf{Adv}^{\text{ind-cpa}}_{\texttt{PKE},\mathcal{A}}(k) \leq \mathbf{Adv}^{\text{ddh}}_{\texttt{GpGen},\mathcal{R}}(k)$ which concludes the proof. □

2.2 Plaintext Awareness in the ROM

The notion of plaintext awareness was first suggested in [BR94] to capture the idea that the only way that an adversary can produce a valid ciphertext is to apply the encryption algorithm to the public key and a message. The motivation was that IND-CPA-security coupled with plaintext awareness should yield IND-CCA2-security, since this property would make the decryption oracle available to an IND-CCA2 adversary useless. The original definition in [BR94], which was formalized in the ROM, was found too weak to imply IND-CCA2-security, and was later adequately refined in [BDPR98]. Providing a satisfactory definition for the standard model turned out to be more subtle and was achieved in [BP04]. Though it was initially thought as a simple tool geared towards proofs of IND-CCA2-security, intrinsic motivations for studying plaintext awareness were later proposed, in particular in order to securely instantiate the ideal encryption functions of the Dolev-Yao model [DY83, HLM03], or to provide deniability to key exchange protocols [RGK06].

In this work, we use the definition of plaintext awareness in the ROM introduced in [BDPR98] and we refer to this definition as ROM-PA-security. This definition involves two types of algorithms: a ciphertext creator $\mathcal{C}$ and a plaintext extractor $\mathcal{P}$. The ciphertext creator is given a public key $\texttt{pk}$ and has access to the random oracle $\boldsymbol{H}$ and to the encryption algorithm $\texttt{PKE.Enc}^{\boldsymbol{H}}_{\texttt{pk}}$. All queries of $\mathcal{C}$ to the random oracle and corresponding answers are recorded in a list $L_{\boldsymbol{H}}$. All answers (ciphertexts) received from the encryption oracle are recorded in a list L_ψ (the corresponding plaintexts are *not* recorded). $\mathcal{C}$ outputs a ciphertext $\psi \notin L_\psi$. We write $(L_{\boldsymbol{H}}, L_\psi, \psi) \leftarrow \texttt{run}\ \mathcal{C}^{\boldsymbol{H},\texttt{PKE.Enc}^{\boldsymbol{H}}_{\texttt{pk}}}(\texttt{pk})$. The plaintext extractor $\mathcal{P}$ takes as input $(L_{\boldsymbol{H}}, L_\psi, \psi, \texttt{pk})$ and aims at returning the plaintext corresponding to ψ.

Definition 2 (ROM-PA). *Let* $\texttt{PKE} = (\texttt{PKE.Kg}, \texttt{PKE.Enc}, \texttt{PKE.Dec})$ *be an encryption scheme.* $\texttt{PKE}$ *is said to be* secure in the sense of plaintext awareness in the ROM *(ROM-PA-secure) if there is a PPT algorithm* $\mathcal{P}$ *(the* plaintext extractor*) such that for any PPT ciphertext creator* $\mathcal{C}$*, the failure probability of* $\mathcal{P}$ *relatively to* $\mathcal{C}$*, defined as:*

$$\texttt{Fail}^{\text{pa}}_{\texttt{PKE},\mathcal{P},\mathcal{C}}(k) = \Pr\left[(\texttt{pk}, \texttt{sk}) \leftarrow \texttt{PKE.Kg}(1^k); (L_{\boldsymbol{H}}, L_\psi, \psi) \leftarrow \texttt{run}\ \mathcal{C}^{\boldsymbol{H},\texttt{PKE.Enc}^{\boldsymbol{H}}_{\texttt{pk}}}(\texttt{pk}) : \mathcal{P}(L_{\boldsymbol{H}}, L_\psi, \psi, \texttt{pk}) \neq \texttt{PKE.Dec}^{\boldsymbol{H}}_{\texttt{sk}}(\psi)\right] ,$$

is a negligible function (the probability is taken over the random tape of all algorithms and the answers of the random oracle).

One may wonder why the ciphertext creator is given access to an encryption oracle since it can encrypt plaintexts by itself using the public key. However, this reflects the fact that an adversary may obtain ciphertexts it has not encrypted by itself (in particular, this is the case of the challenge ciphertext in an IND-CCA2 security experiment), in which case it may not necessarily know the corresponding random oracle queries (which are consequently not listed in $L_{\boldsymbol{H}}$). See [Sho02, BDPR98] for a detailed discussion.

Bellare *et al.* [BDPR98] showed the following theorem.

Theorem 2 ([BDPR98]). *If a PKE scheme is both IND-CPA-secure and ROM-PA-secure, then it is IND-CCA2-secure.*

They also showed that ROM-PA-security is strictly stronger than IND-CCA2-security: there exist IND-CCA2-secure PKE schemes that are not ROM-PA-secure (provided IND-CCA2-secure PKE schemes exist at all).

Note that in the above definition of ROM-PA-security, the plaintext extractor is not given access to the random oracle: it must work with the list of random oracle queries of the ciphertext creator. If one allows the plaintext creator to freely access the random oracle, one loses the intuitively appealing constraint for the plaintext extractor to work given only the view of the ciphertext creator. However, for the results of Appendix B, we will need this relaxation of the definition, and will call ROM-PA'-security the resulting property. Fortunately, it can be checked that the proof of Theorem 2 can be straightforwardly transposed to ROM-PA'-security, namely IND-CPA-security plus ROM-PA'-security implies IND-CCA2-security.

Though the ElGamal encryption scheme does not use the ROM, it is clear that it cannot satisfy any notion of plaintext awareness since an adversary can simply output a random pair $(Y, R) \leftarrow_{\$} \mathbb{G}^2$, which implies that a plaintext extractor should be able to break the one-wayness of the scheme, which holds under the CDH assumption. An attempt to make ElGamal encryption plaintext-aware is to add a Schnorr signature, resulting in what we call the Schnorr-Signed ElGamal (SS-EG) encryption scheme [TY98, Jak98]. Let $\boldsymbol{H_c} : \mathbb{G}^3 \to \mathbb{Z}_p$ be a random oracle. It is defined in Figure 2.

SS-EG PKE scheme

$\mathtt{PKE.Kg}(1^k)$	$\mathtt{PKE.Enc}(\mathtt{pk} = X, M)$	$\mathtt{PKE.Dec}(\mathtt{sk} = x, \psi)$
$(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k)$	$r, a \leftarrow_{\$} \mathbb{Z}_p^*$	Parse ψ as (Y, R, c, s)
$x \leftarrow_{\$} \mathbb{Z}_p^*$; $X := G^x$	$R := G^r$; $R' := X^r$	$R' := R^x$
$\mathtt{sk} := x$; $\mathtt{pk} := X$	$Y := MR'$	$A := G^s R^{-c}$
Return $(\mathtt{sk}, \mathtt{pk})$	$A := G^a$	if $\boldsymbol{H_c}(Y, R, A) \neq c$
	$c := \boldsymbol{H_c}(Y, R, A)$	Return $\perp$
	$s := a + cr \mod p$	Return $M := Y/R'$
	Return $\psi := (Y, R, c, s)$	

Fig. 2. The SS-EG encryption scheme.

As a warm-up for the proof of Theorem 4, we show below that this scheme inherits IND-CPA-security in the ROM under the DDH assumption from basic ElGamal encryption.[10] However, as discussed in the introduction, there is no known proof that this scheme is IND-CCA2-secure in a

[10] Note that in the standard model, one can easily come with instantiations of $\boldsymbol{H_c}$ that ruin even IND-CPA-security of the scheme.

weaker model than the combination of the ROM and the GGM. In Section 4, we prove that this scheme is in fact provably *not* ROM-PA-secure under the CDH assumption.

Theorem 3. *Assuming that the DDH problem is hard relatively to* `GpGen`*, the SS-EG encryption scheme in IND-CPA-secure in the ROM.*

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an IND-CPA adversary against SS-EG making at most q_h random oracle queries. We proceed through a sequence of games as in [BR06]. Each game is a collection of the following procedures:

- **Init** is called by the adversary at the beginning of the game; it runs the key generation algorithm and returns the public key; it also initializes a hashtable H for keeping track of the random oracle queries;
- $\boldsymbol{H_c}$ simulates the random oracle;
- **EncChal**(M_0, M_1) must be called once by the adversary, and returns the challenge ciphertext for the IND-CPA security experiment;
- **Finalize** is called at the end of the game by the adversary when it outputs its guess b', and the output of this procedure is the output of the game $\Gamma^{\mathcal{A}}$.

When a game uses a flag `Bad`, it is always initialized to **false**. The games are defined in Table 3.

We start from game Γ_0 which is simply the IND-CPA security experiment defined in Appendix A. By definition:

$$\mathbf{Adv}^{\text{ind-cpa}}_{\text{SS-EG},\mathcal{A}}(k) = \left|\Pr[\Gamma_0^{\mathcal{A}} = 1] - \frac{1}{2}\right| \ .$$

In game Γ_1, we introduce a flag `Bad` that may be set to **true** at line 13. We also move lines 15 and 17 up in the code. Since R is random and independent of values in hashtable H, one has $\Pr_{\Gamma_1}[\texttt{Bad} = \textbf{true}] \leq q_h/p$. One can check that games Γ_0 and Γ_1 are identical until `Bad` is set to **true**, hence:

$$\left|\Pr[\Gamma_0^{\mathcal{A}} = 1] - \Pr[\Gamma_1^{\mathcal{A}} = 1]\right| \leq \frac{q_h}{p} \ .$$

In game Γ_2, we modify lines 8 to 11. It is easy to see that games Γ_1 and Γ_2 are identical. Namely, in game Γ_2 these lines are equivalent to the following instructions:

$c \leftarrow_{\$} \mathbb{Z}_p$
$s \leftarrow_{\$} \mathbb{Z}_p$
$a := s - rc \mod p$
$A := G^a$

which makes it clear that (r, c, s, a, A) is identically distributed in both games. Hence:

$$\Pr[\Gamma_1^{\mathcal{A}} = 1] = \Pr[\Gamma_2^{\mathcal{A}} = 1] \ .$$

In game Γ_3, we simply change the way R' is generated: it is now randomly drawn. Consider the following distinguisher $\mathcal{B}$ for the DDH problem. It is given a tuple (X, R, R') as input, runs $\mathcal{A}$, and simulates a game similar to games Γ_2 and Γ_3 where it uses X directly instead of line 4 of **Init** and R and R' instead of lines 5 and 6 of **EncChal**. Clearly, when (X, R, R') is a DDH tuple, game Γ_2 is perfectly simulated by $\mathcal{B}$, whereas when (X, R, R') is a random tuple, game Γ_3 is perfectly simulated. Hence:

$$\left|\Pr[\Gamma_2^{\mathcal{A}} = 1] - \Pr[\Gamma_3^{\mathcal{A}} = 1]\right| \leq \mathbf{Adv}^{\text{ddh}}_{\texttt{GpGen},\mathcal{B}}(k) \ .$$

Table 3. Games used in the proof of Theorem 3. Differences from game to game are highlighted in gray.

Game Γ_0:

procedure EncChal(M_0, M_1):
$b \leftarrow_\$ \{0,1\}$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
$R' := X^r$
$Y := M_b R'$
$a \leftarrow_\$ \mathbb{Z}_p^*$

$A := G^a$
if $(Y, R, A) \in H$ **then**:
  $c := H(Y, R, A)$
else $\backslash\backslash (Y, R, A) \notin H$
  $c \leftarrow_\$ \mathbb{Z}_p$
  $H(Y, R, A) := c$
$s := a + cr \mod p$
return (Y, R, c, s)

Game Γ_1:

procedure EncChal(M_0, M_1):
$b \leftarrow_\$ \{0,1\}$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
$R' := X^r$
$Y := M_b R'$
$a \leftarrow_\$ \mathbb{Z}_p^*$
$c \leftarrow_\$ \mathbb{Z}_p$
$s := a + cr \mod p$
$A := G^a$
if $(Y, R, A) \in H$ **then**:
  Bad := **true**
else $\backslash\backslash (Y, R, A) \notin H$

  $H(Y, R, A) := c$

return (Y, R, c, s)

Game Γ_2:

procedure EncChal(M_0, M_1):
$b \leftarrow_\$ \{0,1\}$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
$R' := X^r$
$Y := M_b R'$

$c \leftarrow_\$ \mathbb{Z}_p$
$s \leftarrow_\$ \mathbb{Z}_p$
$A := G^s R^{-c}$
if $(Y, R, A) \in H$ **then**:
  Bad := **true**
else $\backslash\backslash (Y, R, A) \notin H$

  $H(Y, R, A) := c$

return (Y, R, c, s)

Game Γ_3:

procedure EncChal(M_0, M_1):
$b \leftarrow_\$ \{0,1\}$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
$R' \leftarrow_\$ \mathbb{G}$
$Y := M_b R'$

$c \leftarrow_\$ \mathbb{Z}_p$
$s \leftarrow_\$ \mathbb{Z}_p$
$A := G^s R^{-c}$
if $(Y, R, A) \in H$ **then**:
  Bad := **true**
else $\backslash\backslash (Y, R, A) \notin H$

  $H(Y, R, A) := c$

return (Y, R, c, s)

Game Γ_4:

procedure EncChal(M_0, M_1):
$b \leftarrow_\$ \{0,1\}$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$

$Y \leftarrow_\$ \mathbb{G}$

$c \leftarrow_\$ \mathbb{Z}_p$
$s \leftarrow_\$ \mathbb{Z}_p$
$A := G^s R^{-c}$
if $(Y, R, A) \in H$ **then**:
  Bad := **true**
else $\backslash\backslash (Y, R, A) \notin H$

  $H(Y, R, A) := c$

return (Y, R, c, s)

Games Γ_0 to Γ_4:

procedure Init:
$x \leftarrow_\$ \mathbb{Z}_p^*$; $X := G^x$
hashtable $H := \emptyset$
return X

procedure $\boldsymbol{H_c}(Y, R, A)$:
if $(Y, R, A) \notin H$ **then**
  $c \leftarrow_\$ \mathbb{Z}_p$
  $H(Y, R, A) := c$
return $H(Y, R, A)$

procedure Finalize(b'):
return $b' = b$

Finally, in game Γ_4, we replace lines 6 and 7 and simply draw Y at random. Clearly, games Γ_3 and Γ_4 are identical since R' is never used later in the code of these games. Moreover, the view of $\mathcal{A}$ in game Γ_4 is independent of the bit b drawn by **EncChal**, hence:

$$\Pr[\Gamma_3^{\mathcal{A}} = 1] = \Pr[\Gamma_4^{\mathcal{A}} = 1] = \frac{1}{2} \ .$$

Combining all of the above, we obtain:

$$\begin{aligned}\mathbf{Adv}^{\text{ind-cpa}}_{\text{SS-EG},\mathcal{A}}(k) &= \left|\Pr[\Gamma_0^{\mathcal{A}} = 1] - \Pr[\Gamma_3^{\mathcal{A}} = 1]\right| \\ &\leq \mathbf{Adv}^{\text{ddh}}_{\texttt{GpGen},\mathcal{B}}(k) + \frac{q_h}{p} \ .\end{aligned}$$

This concludes the proof. □

3 Chaum-Pedersen-Signed ElGamal Encryption

In this section, we describe our modification of the SS-EG encryption scheme and analyze its security. We name the new scheme Chaum-Pedersen-Signed ElGamal (CPS-EG for short) encryption. The change is quite small: we simply add two elements, $R' = R^x$ and $A' = A^x$, in the inputs to the random oracle $\boldsymbol{H_c}$ when computing the challenge c for the signature. This corresponds to moving from a proof of knowledge of the discrete logarithm $r = \texttt{DLog}_G(R)$ to a Chaum-Pedersen [CP92] proof of *equality* of discrete logarithms $\texttt{DLog}_G(R) = \texttt{DLog}_X(R')$. The scheme uses a random oracle $\boldsymbol{H_c} : \mathbb{G}^5 \to \mathbb{Z}_p$. It is defined in Figure 3. Note that the signature added to the ciphertext is the pair (A, s) rather than (c, s) (*i.e.* the usual Fiat-Shamir signature consisting of the commitment and the response, rather than the optimized variant consisting of the challenge and the response), see Remark 1 below.

CPS-EG PKE scheme

$\texttt{PKE.Kg}(1^k)$	$\texttt{PKE.Enc}(\texttt{pk} = X, M)$	$\texttt{PKE.Dec}(\texttt{sk} = x, \psi)$
$(\mathbb{G}, p, G) \leftarrow \texttt{GpGen}(1^k)$	$r, a \leftarrow_\$ \mathbb{Z}_p^*$	Parse ψ as (Y, R, A, s)
$x \leftarrow_\$ \mathbb{Z}_p^*$; $X := G^x$	$R := G^r$; $R' := X^r$	$R' := R^x$; $A' := A^x$
$\texttt{sk} := x$; $\texttt{pk} := X$	$Y := MR'$	$c := \boldsymbol{H_c}(Y, R, R', A, A')$
Return $(\texttt{sk}, \texttt{pk})$	$A := G^a$; $A' := X^a$	if $G^s \neq AR^c$ or $X^s \neq A'R'^c$
	$c := \boldsymbol{H_c}(Y, R, R', A, A')$	Return $\perp$
	$s := a + cr \mod p$	Return $M := Y/R'$
	Return $\psi := (Y, R, A, s)$	

Fig. 3. The CPS-EG encryption scheme.

Note that the correctness of a ciphertext cannot be checked publicly (*i.e.* without knowledge of the secret key x). As for the Schnorr-Signed variant, the scheme remains IND-CPA-secure under the DDH assumption for GpGen.

Theorem 4. *Assume that the DDH problem is hard for* GpGen*. Then the CPS-EG encryption scheme is IND-CPA-secure in the ROM.*

Table 4. Games used in the proof of Theorem 4. Differences from game to game are highlighted in gray. The symbol $*$ denotes any group element.

Game Γ_0:	**Game** Γ_1:	**Game** Γ_2:
procedure EncChal(M_0, M_1):	**procedure EncChal**(M_0, M_1):	**procedure EncChal**(M_0, M_1):
$b \leftarrow_\$ \{0,1\}$	$b \leftarrow_\$ \{0,1\}$	$b \leftarrow_\$ \{0,1\}$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
$R' := X^r$	$R' := X^r$	$R' := X^r$
$Y := M_b R'$	$Y := M_b R'$	$Y := M_b R'$
$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$
	$c \leftarrow_\$ \mathbb{Z}_p$	
	$s := a + cr \mod p$	$s \leftarrow_\$ \mathbb{Z}_p$
$A' := X^a$	$A' := X^a$	
if $(Y, R, R', A, A') \in H$ **then**:	**if** $(*, *, *, *, A') \in H$ **then**:	
$c := H(Y, R, R', A, A')$	Bad $:=$ **true**	
else $\lozenge(Y, R, R', A, A') \notin H$	**else** $\lozenge(Y, R, R', A, A') \notin H$	
$c \leftarrow_\$ \mathbb{Z}_p$		
$H(Y, R, R', A, A') := c$	$H(Y, R, R', A, A') := c$	
$s := a + cr \mod p$		
return (Y, R, A, s)	**return** (Y, R, A, s)	**return** (Y, R, A, s)
procedure Finalize(b'):	**procedure Finalize**(b'):	**procedure Finalize**(b'):
		$A' := X^a$
		if $(*, *, *, *, A') \in H$ **then**:
		Bad $:=$ **true**
return $b = b'$	**return** $b = b'$	**return** $b = b'$

Game Γ_3:	**Game** Γ_4:	**Games** Γ_0 **to** Γ_4:
procedure EncChal(M_0, M_1):	**procedure EncChal**(M_0, M_1):	**procedure Init**:
$b \leftarrow_\$ \{0,1\}$	$b \leftarrow_\$ \{0,1\}$	$x \leftarrow_\$ \mathbb{Z}_p^*$; $X := G^x$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	hashtable $H := \emptyset$
$R' \leftarrow_\$ \mathbb{G}$		**return** X
$Y := M_b R'$	$Y \leftarrow_\$ \mathbb{G}$	
$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	**procedure** $\boldsymbol{H_c}(Y, R, R', A, A')$:
$s \leftarrow_\$ \mathbb{Z}_p$	$s \leftarrow_\$ \mathbb{Z}_p$	**if** $(Y, R, R', A, A') \notin H$ **then**
return (Y, R, A, s)	**return** (Y, R, A, s)	$c \leftarrow_\$ \mathbb{Z}_p$
		$H(Y, R, R', A, A') := c$
procedure Finalize(b'):	**procedure Finalize**(b'):	**return** $H(Y, R, R', A, A')$
$A' := X^a$	$A' := X^a$	
if $(*, *, *, *, A') \in H$ **then**:	**if** $(*, *, *, *, A') \in H$ **then**:	
Bad $:=$ **true**	Bad $:=$ **true**	
return $b = b'$	**return** $b = b'$	

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an IND-CPA adversary against CPS-EG making at most q_h random oracle queries. We proceed through a sequence of games as in the proof of Theorem 3 (refer to this proof for notations). The games are defined in Table 4.

We start from game Γ_0 which is simply the IND-CPA security experiment defined in Appendix A. By definition:

$$\mathbf{Adv}^{\text{ind-cpa}}_{\text{CPS-EG},\mathcal{A}}(k) = \left|\Pr[\Gamma_0^{\mathcal{A}} = 1] - \frac{1}{2}\right| \ .$$

In game Γ_1, we introduce a flag `Bad` that may be set to **true** at line 13 if H contains any entry $(*, *, *, *, A')$ where $*$ denotes any group element. We also move lines 15 and 17 up in the code. Since A' is random and independent of values in hashtable H, one has $\Pr_{\Gamma_1}[\texttt{Bad} = \textbf{true}] \leq q_h/p$. One can check that games Γ_0 and Γ_1 are identical until `Bad` is set to **true**, hence:

$$\left|\Pr[\Gamma_0^{\mathcal{A}} = 1] - \Pr[\Gamma_1^{\mathcal{A}} = 1]\right| \leq \frac{q_h}{p} \ .$$

In game Γ_2, s is now randomly drawn, and we don't program the random oracle anymore. Instead, we move lines 11, 12 and 13 of **EncChal** to procedure **Finalize**, *i.e.* we check during procedure **Finalize** whether there has been any query of the form $(*, *, *, *, A')$ to the random oracle, where $*$ denotes any group element. Consider the game Γ_1' which is identical to game Γ_1 except that the programming of the random oracle at line 16 is deferred to the (possible) point in the game where the adversary makes the query $\boldsymbol{H_c}(Y, R, R', A, A')$, at which point the flag `Bad` is also set to **true**. Clearly this does not change the output distribution of the game and $\Pr[\Gamma_1^{\mathcal{A}} = 1] = \Pr[\Gamma_1'^{\mathcal{A}} = 1]$. Note that we can also set the flag `Bad` to **true** in game Γ_2 as soon as a query $\boldsymbol{H_c}(*, *, *, *, A')$ occurs. It is now easy to see that games Γ_1' and Γ_2 are identical until `Bad` is set to true, since when `Bad` remains **false** in Γ_1' the view of the adversary is independent of c and hence s looks perfectly random as in Γ_2. Hence:

$$\left|\Pr[\Gamma_1^{\mathcal{A}} = 1] - \Pr[\Gamma_2^{\mathcal{A}} = 1]\right| \leq \Pr_{\Gamma_2}[\texttt{Bad} = \textbf{true}] \ .$$

We now upper bound the probability that `Bad` is set to **true** in Γ_2. For this, we construct an algorithm $\mathcal{B}$ for the CDH problem as follows. $\mathcal{B}$ is given (X, A) where $X = G^x$ and $A = G^a$ as input. It runs $\mathcal{A}$, and simulates game Γ_2 using X and A directly instead of lines 4 of procedure **Init** and line 8 of **EncChal**. When $\mathcal{A}$ finishes, $\mathcal{B}$ picks a random tuple $(*, *, *, *, Z) \in H$ that has been queried to the random oracle by the adversary $\mathcal{A}$ and outputs Z as its guess for G^{ax}. Then, conditioned on `Bad` being true, $\mathcal{B}$ is successful with probability at least $1/q_h$, so that the success probability of this algorithm satisfies:

$$\mathbf{Adv}^{\text{cdh}}_{\texttt{GpGen},\mathcal{B}}(k) \geq \Pr_{\Gamma_2}[\texttt{Bad} = \textbf{true}]/q_h \ .$$

Hence:

$$\left|\Pr[\Gamma_1^{\mathcal{A}} = 1] - \Pr[\Gamma_2^{\mathcal{A}} = 1]\right| \leq q_h \mathbf{Adv}^{\text{cdh}}_{\texttt{GpGen},\mathcal{B}}(k) \ .$$

In game Γ_3, we simply change the way R' is generated: it is now randomly drawn. Exactly as in the proof of Theorem 3, there is an algorithm $\mathcal{B}'$ for the DDH problem such that:

$$\left|\Pr[\Gamma_2^{\mathcal{A}} = 1] - \Pr[\Gamma_3^{\mathcal{A}} = 1]\right| \leq \mathbf{Adv}^{\text{ddh}}_{\texttt{GpGen},\mathcal{B}'}(k) \ .$$

Finally, in game Γ_4, we replace lines 6 and 7 and simply draw Y at random. Clearly, games Γ_3 and Γ_4 are identical since R' is never used later in the code of these games. Moreover, the view of $\mathcal{A}$ in game Γ_4 is independent of bit b, hence:

$$\Pr[\Gamma_3^{\mathcal{A}} = 1] = \Pr[\Gamma_4^{\mathcal{A}} = 1] = \frac{1}{2} \ .$$

Combining all the above, we obtain:

$$\begin{aligned}\mathbf{Adv}^{\text{ind-cpa}}_{\text{CPS-EG},\mathcal{A}}(k) &= \left|\Pr[\Gamma_0^{\mathcal{A}} = 1] - \Pr[\Gamma_3^{\mathcal{A}} = 1]\right| \\ &\leq q_h \mathbf{Adv}^{\text{cdh}}_{\texttt{GpGen},\mathcal{B}}(k) + \mathbf{Adv}^{\text{ddh}}_{\texttt{GpGen},\mathcal{B}'}(k) + \frac{q_h}{p} \ .\end{aligned}$$

This concludes the proof. □

Remark 1. In a previous version of this paper, the ciphertext produced by the encryption algorithm was (Y, R, c, s) as in SS-EG rather than (Y, R, A, s). However this yields an insecure variant of the scheme, as was pointed out to us by Poettering [Poe13]. Indeed, consider the following IND-CPA adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. $\mathcal{A}_1$ receives the public key $X = G^x$, and simply outputs two arbitrary messages M_0 and M_1. $\mathcal{A}_2$ then receives the challenge ciphertext (Y, R, c, s), where $R = G^r$, $R' = X^r$, $Y = M_b R'$, and c and s are computed by drawing $a \leftarrow_\$ \mathbb{Z}_p$, and computing $A = G^a$, $A' = X^a$, $c = \boldsymbol{H_c}(Y, R, R', A, A')$ and $s = a + cr \bmod p$. Then $\mathcal{A}_2$ proceeds as follows: it computes $A = G^s R^{-c}$ as well as candidate values $R'_0 = Y/M_0$, $A'_0 = X^s R_0'^{-c}$, and $R'_1 = Y/M_1$, $A'_1 = X^s R_1'^{-c}$. Finally it computes $c_0 = \boldsymbol{H_c}(Y, R, R'_0, A, A'_0)$ and $c_1 = \boldsymbol{H_c}(Y, R, R'_1, A, A'_1)$, and checks whether c_0 or c_1 matches c, returning b' such that $c_{b'} = c$. This guess is correct with overwhelming probability.

The intuition regarding why the CPS-EG scheme is ROM-PA-secure is simple: since R' is now included in the random oracle queries, the plaintext extractor will directly be able to decrypt Y once it has located the corresponding query. Namely, upon reception of a ciphertext (Y, R, A, s), it can look throughout the list of random oracle queries to find all queries of the form (Y, R, R'_i, A, A'_i) with corresponding answers c_i, and check whether $G^s = AR^{c_i}$ and $X^s = A'_i R_i'^{c_i}$. The uniqueness of a query passing both checks is ensured by the following lemma (which is the core of the proof of soundness of the Chaum-Pedersen protocol [CP92], and also the base of the "twinning" technique of Cash *et al.* [CKS08]).

Lemma 1. *Fix $X = G^x$, $x \in \mathbb{Z}_p^*$. Let $R = G^r, R', A = G^a, A' \in \mathbb{G}$ be four group elements such that $r, a \neq 0 \bmod p$, and $R' \neq R^x$ or $A' \neq A^x$. Then there is at most one integer $c \in \mathbb{Z}_p$ such that there exists $s \in \mathbb{Z}_p$ satisfying both $G^s = AR^c$ and $X^s = A'R'^c$.*

Proof. Denote $R' = R^y$ and $A' = A^z$, and assume that there exists (s_1, c_1) and (s_2, c_2) with $c_1 \neq c_2$ satisfying $G^{s_i} = AR^{c_i}$ and $X^{s_i} = A'R'^{c_i}$ for $i = 1, 2$. Then $G^{xs_i} = A^x R^{xc_i} = A'R'^{c_i} = A^z R^{yc_i}$, which implies $a(x - z) = rc_i(y - x) \bmod p$. Hence if $y \neq x$, then $c_1 = c_2 = a(x - z)/r(y - x) \bmod p$, contradicting the assumption that $c_1 \neq c_2$. If $y = x$ then $z = x$ (since $a \neq 0 \bmod p$) which contradicts the assumption that $R' \neq R^x$ or $A' \neq A^x$. □

We now give the formal proof that the scheme is ROM-PA-secure.

Theorem 5. *The CPS-EG encryption scheme is ROM-PA-secure. Hence, it is IND-CCA2-secure in the ROM under the assumption that DDH is hard for* `GpGen`.

Proof. Consider a ciphertext creator $\mathcal{C}$ making at most q_h random oracle queries and q_e queries to the encryption oracle. Let $(L_{\boldsymbol{H}}, L_\psi, \psi)$ be the output of a run of $\mathcal{C}$ on public key $\texttt{pk} = X = G^x$, where $\psi = (Y, R, A, s)$. The plaintext extractor $\mathcal{P}$ proceeds as follows: it looks throughout $L_{\boldsymbol{H}}$ for all queries of the form $(Y, R, *, A, *)$, where $*$ denotes any group element. If there is no such query, it returns $\perp$ (meaning that the ciphertext is invalid). Otherwise, denote (Y, R, R'_i, A, A'_i) all queries of this form and c_i the corresponding answers. For each i, $\mathcal{P}$ checks whether $G^s = AR^{c_i}$ and $X^s = A'_i R_i'^{c_i}$. If there is no such query, it returns $\perp$. If there is more than one such query, the plaintext extractor

aborts. Otherwise, denoting (Y, R, R', A, A') the unique query for which both checks passed, $\mathcal{P}$ returns $M = Y/R'$ as the plaintext.

We now analyze the failure probability of the plaintext extractor. For this, we define a bad event as follows: event $\texttt{Bad}$ happens if there exists some query (Y, R, R'_i, A, A'_i) in $L_{\boldsymbol{H}}$ such that $R'_i \neq R^x$ or $A'_i \neq A^x$, and the corresponding answer c_i passes both checks $G^s = AR^{c_i}$ and $X^s = A'_i R'^{c_i}_i$. Since Lemma 1 ensures that for each query there is at most one possible answer $c'_i \in \mathbb{Z}_p$ such that both checks may pass, we see that:

$$\Pr[\texttt{Bad}] \leq \frac{q_h}{p} \ .$$

We now consider three distinct cases that may occur.

1. **Case 1**: There is no query of the form $(Y, R, *, A, *)$ in $L_{\boldsymbol{H}}$, or none of them passes the checks. Here, we distinguish two sub-cases. The first one happens if there exists a ciphertext $(Y, R, A, s') \in L_\psi$ with $s' \neq s$ (since $\psi \notin L_\psi$). Since (Y, R, A, s') is necessarily valid, we see that the challenge ciphertext (Y, R, A, s) is necessarily invalid and $\mathcal{P}$ is correct in returning $\perp$. The second sub-case happens if there is no ciphertext $(Y, R, A, *) \in L_\psi$. In this case, the query (Y, R, R^x, A, A^x) cannot have been issued to the random oracle during any of the q_e calls of the ciphertext creator to $\texttt{PKE.Enc}^{\boldsymbol{H}}_{\texttt{pk}}$. Hence, if this query is not in $L_{\boldsymbol{H}}$, then the view of $\mathcal{C}$ and $\mathcal{P}$ is independent of the value of $\boldsymbol{H_c}$ at this point. Thus, the ciphertext is valid with probability at most $1/p$. Hence, by returning $\perp$, the plaintext extractor errs with probability at most $1/p$.
2. **Case 2**: There are two distinct queries (Y, R, R'_i, A, A'_i) and (Y, R, R'_j, A, A'_j) in $L_{\boldsymbol{H}}$ which pass the checks. Then necessarily one of these two queries (say query i) is such that $R'_i \neq R^x$ or $A'_i \neq A^x$. But this implies that $\texttt{Bad}$ has happened, so that this case occurs with probability less than q_h/p.
3. **Case 3**: There is a unique query (Y, R, R', A, A') such that both checks pass. If $R' = R^x$ and $A' = A^x$, then it is clear that the plaintext extractor returns the plaintext that would be returned by $\texttt{PKE.Dec}^{\boldsymbol{H}}_{\texttt{sk}}$. Else, $R' \neq R^x$ or $A' \neq A^x$, and this implies that $\texttt{Bad}$ must have happened. Hence the plaintext extractor returns an incorrect plaintext with probability at most q_h/p.

Collecting all cases, we have:

$$\begin{aligned}\Pr[\mathcal{P} \text{ errs}] &\leq \Pr[\mathcal{P} \text{ errs}|\text{Case 1}] + \Pr[\text{Case 2}] + \Pr[\mathcal{P} \text{ errs}|\text{Case 3}] \\ &\leq \frac{2q_h + 1}{p} \ ,\end{aligned}$$

which is negligible in the security parameter. The scheme is ROM-PA and it follows from Theorems 2 and 4 that the scheme is IND-CCA2-secure in the ROM under the assumption that DDH is hard for $\texttt{GpGen}$. □

We note that CPS-EG seems to be the simplest IND-CCA2-secure scheme in the ROM that retains some kind of homomorphic property. Namely, given two ciphertexts (Y_0, R_0, A_0, s_0) and (Y_1, R_1, A_1, s_1) corresponding respectively to plaintexts M_0 and M_1, one can compute the first two elements of the ciphertext for $M_0 M_1$ as for basic ElGamal encryption. It is however impossible to "sign" the new ciphertext without knowledge of the random values used to encrypt M_0 and M_1. This property appears to be useful in e-voting applications [Wik08, BCP+11].

4 Minimality of CPS-EG Regarding Plaintext Awareness

In this section, we consider whether the CPS-EG encryption scheme remains ROM-PA-secure when some inputs are removed from the call to the random oracle $\boldsymbol{H_c}$. We are able to show that under an adequate assumption, the resulting scheme *cannot* be ROM-PA-secure. Namely:

– If we remove R' and A', we exactly recover the SS-EG scheme, and we can show that the scheme is not ROM-PA-secure under the CDH assumption.
– If we remove one single element among R, R', A, or A', then the scheme is not ROM-PA-secure under the DDH assumption. Moreover, removing Y trivially makes the scheme malleable, while removing A' yields a scheme which is not IND-CPA-secure.

This is captured by the following two theorems.

Theorem 6. *Assume that the CDH problem is hard for* $\texttt{GpGen}$*. Then the SS-EG encryption scheme is not ROM-PA-secure.*

Proof. Assume for contradiction that SS-EG encryption is ROM-PA-secure, and let $\mathcal{P}$ be a plaintext extractor. Note that this extractor must work for any plaintext creator, in particular for the ciphertext creator $\mathcal{C}^*$ working as follows: $\mathcal{C}^*$ draws a random message $M \leftarrow_\$ \mathbb{G}$, and encrypts it honestly. We now build a reduction $\mathcal{R}$ that solves the CDH problem. Let $(\mathbb{G}, p, G) \leftarrow \texttt{GpGen}(1^k)$ and $(X = G^x, R = G^r)$ denote the input to $\mathcal{R}$. $\mathcal{R}$ simulates the run of the ciphertext creator $\mathcal{C}^*$ as follows. It draws two integers $c, s \leftarrow_\$ \mathbb{Z}_p$, a random group element $Y \leftarrow_\$ \mathbb{G}$, computes $A = G^s R^{-c}$, and programs the random oracle with $\boldsymbol{H_c}(Y, R, A) = c$. Then it runs the plaintext extractor with input $L_{\boldsymbol{H_c}} = ((Y, R, A), c)$, $L_\psi = \emptyset$, $\psi = (Y, R, c, s)$, and $\texttt{pk} = X$. It is clear that this is a perfect simulation of a run of $\mathcal{C}^*$ (in particular the ciphertext is valid, and that the simulation of the random oracle is perfect). Hence, with overwhelming probability, $\mathcal{P}$ returns the plaintext M corresponding to ψ, from which $\mathcal{R}$ can compute $G^{rx} = Y/M$. □

Since ROM-PA-security is strictly stronger than IND-CCA2-security, the result above does not imply that SS-EG is not IND-CCA2-secure in the ROM. Would SS-EG be proved IND-CCA2-secure this would yield a natural separation between this notion and ROM-PA-security (the separation provided in [BDPR98] used a rather contrived counter-example, but without any computational assumption). Also, there are many possible ways to weaken the definition of plaintext awareness in the ROM. The theorem above seems to crucially rely on the impossibility for the plaintext extractor to rewind the ciphertext creator or to program the random oracle: it must work online, and can only observe the ciphertext creator queries. In particular, [BP04] proposed weaker notions of PA in the ROM where the plaintext extractor is not black-box (it can depend on the code of the ciphertext creator), and is given the random coins of the ciphertext creator, which enables to rewind it. Exploring whether SS-EG may fulfill such weaker definitions is an interesting open question.

Regarding the minimality of CPS-EG, we have the following result.

Theorem 7. *Consider the CPS-EG scheme where one of the five elements* (Y, R, R', A, A') *is omitted from the inputs the* $\boldsymbol{H_c}$ *when generating* c*. Then one has the following:*

(a) If Y *is omitted, then the resulting scheme is malleable and hence is not IND-CCA2-secure.*
(b) If A' *is omitted, then the resulting scheme is not IND-CPA-secure.*
(c) If R*,* R'*,* A *or* A' *is omitted, then assuming that DDH is hard for* $\texttt{GpGen}$*, the resulting scheme is not ROM-PA-secure.*

Proof. We first show (a): removing Y from the inputs to $\boldsymbol{H_c}$ causes the encryption scheme to be malleable. Suppose (Y, R, A, s) is a valid ciphertext for some message M, then the 4-tuple (YG^z, R, A, s) is a valid ciphertext for the message $M' = MG^z$. As a consequence, the resulting scheme is not IND-CCA2 secure.

We then show (b). Consider the scheme obtained when A' is omitted, namely c is computed as $c := \boldsymbol{H_c}(Y, R, R', A)$. Consider the following IND-CPA attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. $\mathcal{A}_1$ simply outputs

two arbitrary messages M_0 and M_1. $\mathcal{A}_2$ receives the challenge ciphertext (Y, R, A, s). It computes two candidate values $R'_0 = Y/M_0$ and $R'_1 = Y/M_1$, and queries $c_0 = \boldsymbol{H_c}(Y, R, R'_0, A)$ and $c_1 = \boldsymbol{H_c}(Y, R, R'_1, A)$. Then it checks whether $G^s = AR^{c_{b'}}$ for $b' = 0$ and 1, returning b' such that the check succeeds. This adversary has advantage close to 1.

We finally show (c). We focus on the case where R' is omitted first, namely c is computed as $c := \boldsymbol{H_c}(Y, R, A, A')$. Assume for contradiction that the resulting scheme is ROM-PA-secure, and let $\mathcal{P}$ be a plaintext extractor. Consider the two following ciphertext creators $\mathcal{C}^*_{\text{ddh}}$ and $\mathcal{C}^*_{\text{rand}}$. $\mathcal{C}^*_{\text{ddh}}$ simply draws a random message $M \leftarrow_\$ \mathbb{G}$ and encrypts it honestly. $\mathcal{C}^*_{\text{rand}}$ on the other hand behaves as described in Table 5. Note that the ciphertext created by $\mathcal{C}^*_{\text{rand}}$ is invalid with overwhelming probability.

Table 5. Ciphertext creators $\mathcal{C}^*_{\text{ddh}}$ and $\mathcal{C}^*_{\text{rand}}$.

$\mathcal{C}^*_{\text{ddh}}(\mathtt{pk} = X)$:	$\mathcal{C}^*_{\text{rand}}(\mathtt{pk} = X)$:
$M \leftarrow_\$ \mathbb{G}$	$M \leftarrow_\$ \mathbb{G}$
$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
$R' := X^r$	$R' := X^r$
$Y := MR'$	$Y := MR'$
$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$
$A' := X^a$	$A' \leftarrow_\$ \mathbb{G}$
$c := \boldsymbol{H_c}(Y, R, A, A')$	$c := \boldsymbol{H_c}(Y, R, A, A')$
$s := a + cr \bmod p$	$s := a + cr \bmod p$
return (Y, R, A, s)	**return** (Y, R, A, s)

We now build a reduction $\mathcal{R}$ solving the DDH problem. Let $(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k)$ and $(X = G^x, A = G^a, A')$ be the input to $\mathcal{R}$. $\mathcal{R}$ proceeds as follows: it draws two integers $c, s \leftarrow_\$ \mathbb{Z}_p$, a random group element $Y \leftarrow_\$ \mathbb{G}$, computes $R = (AG^{-s})^{-1/c}$ (note that $c \neq 0$ with probability $1 - 1/p$), and programs the random oracle with $\boldsymbol{H_c}(Y, R, A, A') = c$. Then it runs the plaintext extractor with input $L_{\boldsymbol{H_c}} = ((Y, R, A, A'), c)$, $L_\psi = \emptyset$, $\psi = (Y, R, A, s)$ and $\mathtt{pk} = X$. If the plaintext extractor outputs any message M, then $\mathcal{R}$ outputs 1. If the plaintext extractor outputs $\perp$, then $\mathcal{R}$ outputs 0. One can check that when (X, A, A') is a DDH tuple, $\mathcal{R}$ perfectly simulates a run of $\mathcal{C}^*_{\text{ddh}}$, whereas when (X, A, A') is a random tuple, $\mathcal{R}$ perfectly simulates a run of $\mathcal{C}^*_{\text{rand}}$. Since in both cases the extractor must return the correct answer with overwhelming probability, $\mathcal{R}$ solves the DDH problem with advantage close to 1.

The reasoning is similar for the cases where R, A, or A' are omitted. For completeness, we describe the corresponding ciphertext creators $\mathcal{C}^*_{\text{rand}}$ in Table 6. □

5 Anonymity and Robustness of CPS-EG

The formal definition of anonymity for a PKE scheme is recalled in Appendix A. Informally, this requires that an adversary cannot distinguish under which public key an (adversarially chosen) message was encrypted. We start with showing that the CPS-EG scheme provides anonymity under CCA2 attacks.

Theorem 8. *Assume that the DDH problem is hard for* `GpGen`*. Then the CPS-EG encryption scheme is ANON-CCA2-secure.*

Table 6. Ciphertext creators $\mathcal{C}^*_{\text{rand}}$ for the cases where R, A, or A' are omitted.

(R omitted)

1 **procedure** $\mathcal{C}^*_{\text{rand}}(\mathtt{pk} = X)$:
2 $M \leftarrow_\$ \mathbb{G}$
3 $r \leftarrow_\$ \mathbb{Z}_p^*$
4 $R' := X^r$
5 $Y := MR'$
6 $A \leftarrow_\$ \mathbb{G}$
7 $a \leftarrow_\$ \mathbb{Z}_p^*$; $A' := X^a$
8 $c := \boldsymbol{H_c}(Y, R', A, A')$
9 $s := a + cr \bmod p$
10 $R := (AG^{-s})^{-1/c}$
11 **return** (Y, R, A, s)

(A omitted)

1 **procedure** $\mathcal{C}^*_{\text{rand}}(\mathtt{pk} = X)$:
2 $M \leftarrow_\$ \mathbb{G}$
3 $r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
4 $R' \leftarrow_\$ \mathbb{G}$
5 $Y := MR'$
6
7 $a \leftarrow_\$ \mathbb{Z}_p^*$; $A' := X^a$
8 $c := \boldsymbol{H_c}(Y, R, R', A')$
9 $s := a + cr \bmod p$
10 $A := G^s R^{-c}$
11 **return** (Y, R, A, s)

(A' omitted)

1 **procedure** $\mathcal{C}^*_{\text{rand}}(\mathtt{pk} = X)$:
2 $M \leftarrow_\$ \mathbb{G}$
3 $r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
4 $R' \leftarrow_\$ \mathbb{G}$
5 $Y := MR'$
6 $a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$
7
8 $c := \boldsymbol{H_c}(Y, R, R', A)$
9 $s := a + cr \bmod p$
10
11 **return** (Y, R, A, s)

Proof. We first show ANON-CPA-security. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an ANON-CPA adversary against CPS-EG making at most q_h random oracle queries. We proceed through a sequence of games as in the proof of Theorem 4 (refer to this proof for notations). The games are defined in Table 7.

We start from game Γ_0 which is simply the ANON-CPA security experiment defined in Appendix A. By definition:

$$\mathbf{Adv}^{\text{anon-cpa}}_{\text{CPS-EG},\mathcal{A}}(k) = \left|\Pr[\Gamma_0^{\mathcal{A}} = 1] - \frac{1}{2}\right| \ .$$

The transitions from game to game are then exactly the same as in the proof of Theorem 4, and we refer to this proof for a detailed analysis. The only difference stands in the way the probability that $\mathtt{Bad}$ is set to true in game Γ_2 is bounded. For this, we construct an algorithm $\mathcal{B}$ for the CDH problem as follows. $\mathcal{B}$ is given (X, A) where $X = G^x$ and $A = G^a$ as input. It runs $\mathcal{A}$, and simulates game Γ_2 by replacing lines 4 and 5 of **Init** as follows: it draws $\alpha_0, \alpha_1 \leftarrow_\$ \mathbb{Z}_p^*$, and computes $X_0 = X^{\alpha_0}$ and $X_1 = X^{\alpha_1}$. It also uses its input A directly instead of line 8 of **EncChal**. When $\mathcal{A}$ finishes, $\mathcal{B}$ takes a random tuple $(*, *, *, *, Z)$ that has been queried to the random oracle by the adversary $\mathcal{A}$ and outputs Z^{1/α_b} as its guess for G^{ax}. Clearly, game Γ_2 is perfectly simulated by $\mathcal{B}$. Then, conditioned on $\mathtt{Bad}$ being true, $\mathcal{B}$ is successful with probability at least $1/q_h$, so that the success probability of this algorithm satisfies:

$$\mathbf{Adv}^{\text{cdh}}_{\mathtt{GpGen},\mathcal{B}}(k) \geq \Pr_{\Gamma_2}[\mathtt{Bad} = \mathbf{true}]/q_h \ .$$

As in the proof of Theorem 4, we obtain:

$$\mathbf{Adv}^{\text{anon-cpa}}_{\text{CPS-EG},\mathcal{A}}(k) \leq q_h \mathbf{Adv}^{\text{cdh}}_{\mathtt{GpGen},\mathcal{B}}(k) + \mathbf{Adv}^{\text{ddh}}_{\mathtt{GpGen},\mathcal{B}'}(k) + \frac{q_h}{p} \ .$$

For the proof of ANON-CCA2-security, we use a composition theorem relying on the plaintext awareness of the scheme. However, the definition of ROM-PA given in Section 2.2 is well-suited to be composed with IND-CPA-security and not with ANON-CPA-security. Hence, we have to slightly modify the definition of ROM-PA into a new notion that we name ROM-ANON-PA. The new definition involves a ciphertext creator $\mathcal{C}$ which is given *two* random public keys $\mathtt{pk}_0$ and $\mathtt{pk}_1$. It has access to the random oracle $\boldsymbol{H}$ and to the two encryption oracles $\mathtt{PKE.Enc}^{\boldsymbol{H}}_{\mathtt{pk}_0}$ and $\mathtt{PKE.Enc}^{\boldsymbol{H}}_{\mathtt{pk}_1}$. All queries of $\mathcal{C}$ to the random oracle and corresponding answers are recorded in a list $L_{\boldsymbol{H}}$. All answers

Table 7. Games used in the proof of Theorem 8. Differences from game to game are highlighted in gray. The symbol $*$ denotes any group element.

	Game Γ_0:	**Game** Γ_1:	**Game** Γ_2:
1			
2	**procedure EncChal**(M):	**procedure EncChal**(M):	**procedure EncChal**(M):
3	$b \leftarrow_\$ \{0,1\}$	$b \leftarrow_\$ \{0,1\}$	$b \leftarrow_\$ \{0,1\}$
4	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$
5	$R' := (X_b)^r$	$R' := (X_b)^r$	$R' := (X_b)^r$
6	$Y := MR'$	$Y := MR'$	$Y := MR'$
7	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$
8		$c \leftarrow_\$ \mathbb{Z}_p$	
9		$s := a + cr \mod p$	$s \leftarrow_\$ \mathbb{Z}_p$
10	$A' := (X_b)^a$	$A' := (X_b)^a$	
11	**if** $(Y, R, R', A, A') \in H$ **then**	**if** $(*, *, *, *, A') \in H$ **then**	
12	$\quad c := H(Y, R, R', A, A')$	$\quad$ Bad $:=$ **true**	
13	**else** $\Vert(Y, R, R', A, A') \notin H$	**else** $\Vert(Y, R, R', A, A') \notin H$	
14	$\quad c \leftarrow_\$ \mathbb{Z}_p$		
15	$\quad H(Y, R, R', A, A') := c$	$\quad H(Y, R, R', A, A') := c$	$\quad H(Y, R, R', A, A') := c$
16	$s := a + cr \mod p$		
17	**return** (Y, R, A, s)	**return** (Y, R, A, s)	**return** (Y, R, A, s)
18			
19	**procedure Finalize**(b'):	**procedure Finalize**(b'):	**procedure Finalize**(b'):
20			$A' := (X_b)^a$
21			**if** $(*, *, *, *, A') \in H$ **then**:
22			$\quad$ Bad $:=$ **true**
23	**return** $b = b'$	**return** $b = b'$	**return** $b = b'$

	Game Γ_3:	**Game** Γ_4:	**Games** Γ_0 **to** Γ_4:
1			
2	**procedure EncChal**(M):	**procedure EncChal**(M):	**procedure Init**:
3	$b \leftarrow_\$ \{0,1\}$	$b \leftarrow_\$ \{0,1\}$	$x_0 \leftarrow_\$ \mathbb{Z}_p^*$, $X_0 := G^{x_0}$
4	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$r \leftarrow_\$ \mathbb{Z}_p^*$; $R := G^r$	$x_1 \leftarrow_\$ \mathbb{Z}_p^*$; $X_1 := G^{x_1}$
5	$R' \leftarrow_\$ \mathbb{G}$		hashtable $H := \emptyset$
6	$Y := MR'$	$Y \leftarrow_\$ \mathbb{G}$	**return** (X_0, X_1)
7	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	$a \leftarrow_\$ \mathbb{Z}_p^*$; $A := G^a$	
8	$s \leftarrow_\$ \mathbb{Z}_p$	$s \leftarrow_\$ \mathbb{Z}_p$	**procedure** $\boldsymbol{H}_c(Y, R, R', A, A')$:
9	**return** (Y, R, A, s)	**return** (Y, R, A, s)	**if** $(Y, R, R', A, A') \notin H$ **then**
10			$\quad c \leftarrow_\$ \mathbb{Z}_p$
11	**procedure Finalize**(b'):	**procedure Finalize**(b'):	$\quad H(Y, R, R', A, A') := c$
12	$A' := (X_b)^a$	$A' := (X_b)^a$	**return** $H(Y, R, R', A, A')$
13	**if** $(*, *, *, *, A') \in H$ **then**:	**if** $(*, *, *, *, A') \in H$ **then**:	
14	$\quad$ Bad $:=$ **true**	$\quad$ Bad $:=$ **true**	
15	**return** $b = b'$	**return** $b = b'$	

(ciphertexts) received from the two encryption oracles are recorded in a list L_ψ (neither the corresponding plaintexts nor which oracle was queried are recorded). $\mathcal{C}$ outputs a ciphertext $\psi \notin L_\psi$ and a bit b. We write $(L_{\boldsymbol{H}}, L_\psi, \psi, b) \leftarrow \texttt{run}\ \mathcal{C}^{\boldsymbol{H},\texttt{PKE.Enc}^{\boldsymbol{H}}_{\texttt{pk}_0,\texttt{pk}_1}}(\texttt{pk}_0, \texttt{pk}_1)$. The plaintext extractor $\mathcal{P}$ takes as input $(L_{\boldsymbol{H}}, L_\psi, \psi, b, \texttt{pk}_0, \texttt{pk}_1)$ and aims at returning the plaintext corresponding to ψ and secret key $\texttt{sk}_b$. PKE is said to be ROM-ANON-PA-secure if there is a PPT algorithm $\mathcal{P}$ (the *plaintext extractor*) such that for any PPT ciphertext creator $\mathcal{C}$, the failure probability of $\mathcal{P}$ relatively to $\mathcal{C}$, defined as:

$$\texttt{Fail}^{\text{anon-pa}}_{\texttt{PKE},\mathcal{P},\mathcal{C}}(k) = \Pr\left[(\texttt{pk}_0, \texttt{sk}_0) \leftarrow \texttt{PKE.Kg}(1^k); (\texttt{pk}_1, \texttt{sk}_1) \leftarrow \texttt{PKE.Kg}(1^k);\right.$$
$$\left.(L_{\boldsymbol{H}}, L_\psi, \psi, b) \leftarrow \texttt{run}\ \mathcal{C}^{\boldsymbol{H},\texttt{PKE.Enc}^{\boldsymbol{H}}_{\texttt{pk}_0,\texttt{pk}_1}}(\texttt{pk}_0, \texttt{pk}_1) \ :\ \mathcal{P}(L_{\boldsymbol{H}}, L_\psi, \psi, b, \texttt{pk}_0, \texttt{pk}_1) \neq \texttt{PKE.Dec}^{\boldsymbol{H}}_{\texttt{sk}_b}(\psi)\right]\ ,$$

is a negligible function (the probability is taken over the random tape of all algorithms and the answers of the random oracle).

One can then show that if a scheme is ANON-CPA-secure and ROM-ANON-PA secure, then it is ANON-CCA2-secure. We sketch the justification of this fact. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an ANON-CCA2 adversary against a PKE scheme. We build an ANON-CPA adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ as follows. $\mathcal{B}_1$ is given two public keys $\texttt{pk}_0$ and $\texttt{pk}_1$, and runs $\mathcal{A}_1$ with the same public keys. It simply relays the random oracle queries of $\mathcal{A}_1$ to its own random oracle, updating a list $L_{\boldsymbol{H}}$ of queries and corresponding answers for the plaintext extractor. When $\mathcal{A}_1$ makes a decryption query for ψ and key b, $\mathcal{B}_1$ runs the plaintext extractor $\mathcal{P}$ with input $(L_{\boldsymbol{H}}, L_\psi = \emptyset, \psi, b, \texttt{pk}_0, \texttt{pk}_1)$ and returns the corresponding output as the answer to the decryption query of $\mathcal{A}_1$. When $\mathcal{A}_1$ outputs a message M, $\mathcal{B}_1$ outputs the same message M, and receives a challenge ψ^*. In the second phase of the security experiment, $\mathcal{B}_2$ runs $\mathcal{A}_2$ on the same challenge ψ^*. As in the first phase, it simply relays the random oracle queries of $\mathcal{A}_2$ to its own random oracle, updating the list $L_{\boldsymbol{H}}$ accordingly. When $\mathcal{A}_2$ makes a decryption query for ψ and key b, $\mathcal{B}_2$ runs the plaintext extractor $\mathcal{P}$ with input $(L_{\boldsymbol{H}}, L_\psi = (\psi^*), \psi, b, \texttt{pk}_0, \texttt{pk}_1)$ and returns the corresponding output as the answer to the decryption query of $\mathcal{A}_2$. Finally, it returns the same output as $\mathcal{A}_2$. The analysis of the advantage of $\mathcal{B}$ can be adapted from the proof of Theorem 4.2 of [BDPR98], and it can be shown to be negligibly close to the one of $\mathcal{A}$.

Finally, it remains to show that CPS-EG indeed satisfies the ROM-ANON-PA notion. This can be done with exactly the same extractor as in the proof of Theorem 5 (one simply has to replace X by the public key $X_b = G^{x_b}$ corresponding to the bit b output by $\mathcal{C}$). The analysis of the success probability must be slightly adapted as follows. Denote $\texttt{pk}_0 = G^{x_0}$ and $\texttt{pk}_1 = G^{x_1}$ the public keys given as input to $\mathcal{C}$. Let $\psi = (Y, R, A, s)$ be the challenge ciphertext output by $\mathcal{C}$, and assume *wlog* that $\mathcal{C}$ asks for the decryption of this ciphertext under secret key $\texttt{sk}_0 = x_0$ ($b = 0$ in the output of $\mathcal{C}$). The analysis of cases 2 and 3 is unchanged. For case 1, only the first sub-case (*i.e.* when there exists some ciphertext $\psi' = (Y, R, A, s') \in L_\psi$ with $s' \neq s$) is slightly modified. If this ciphertext was returned in answer to some encryption query for $\texttt{pk}_0$, then it is necessarily valid and hence the challenge ψ is invalid. If this ciphertext was returned in answer to some encryption query for $\texttt{pk}_1$, then assuming *wlog* that $x_0 \neq x_1$, the random oracle query to $\boldsymbol{H_c}$ made while computing this ciphertext was $(Y, R, R^{x_1}, A, A^{x_1})$. Hence the view of $\mathcal{C}$ and $\mathcal{P}$ is independent of the value of $\boldsymbol{H_c}$ on input $(Y, R, R^{x_0}, A, A^{x_0})$ so that the plaintext extractor errs by returning $\perp$ with probability at most $1/p$. □

The definition of strong robustness is recalled in Appendix A. We will now see that the CPS-EG scheme achieves strong robustness, assuming the following very simple tweak to the scheme (in fact, the same was proposed for Cramer-Shoup encryption in [ABN10]). We define the CPS-EG* scheme exactly as the CPS-EG scheme, with the additional check on decryption that $R \neq 1_{\mathbb{G}}$ (if the check fails, then the decryption algorithm returns $\perp$).

Theorem 9. *Assume $\boldsymbol{H_c}$ is instantiated with a collision-resistant hash function family. Then the CPS-EG* encryption scheme is SROB-CCA-secure.*

Proof. Assume there is an adversary $\mathcal{A}$ with non-negligible advantage in the SROB-CCA game. Let $X_0 = G^{x_0}$ and $X_1 = G^{x_1}$ (where $x_0 \neq x_1$ with high probability) be the two public keys for which the adversary must return a ciphertext that decrypts to a valid plaintext under both secret keys x_0 and x_1 (we can assume that the challenger knows these secret keys, so that it can correctly answer any decryption query made by the adversary). Let $\psi = (Y, R, A, s)$ be the ciphertext returned by $\mathcal{A}$, where $R = G^r$ and $A = G^a$. The validity check under secret key x_0 uses values $R'_0 = R^{x_0}$ and $A'_0 = A^{x_0}$, whereas the validity check under secret key x_1 uses values $R'_1 = R^{x_1}$ and $A'_1 = A^{x_1}$. If the validity check succeeds in both cases, then necessarily $R \neq 1_{\mathbb{G}}$, so that $R'_0 \neq R'_1$. Moreover, denoting $c_0 = \boldsymbol{H_c}(Y, R, R'_0, A, A'_0)$ and $c_1 = \boldsymbol{H_c}(Y, R, R'_1, A, A'_1)$, one has $s = a + c_0 r \bmod p$ and $s = a + c_1 r \bmod p$, so that $c_0 = c_1$. This means that $\mathcal{A}$ succeeds in finding a collision for $\boldsymbol{H_c}$, a contradiction with the assumption of collision-resistance for the hash function family used to instantiate $\boldsymbol{H_c}$. □

Acknowledgments

We are thankful to Bertram Poettering who pointed out to us that a previous version of the CPS-EG scheme was insecure.

References

[ABC+08] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. *Journal of Cryptology*, 21(3):350–391, 2008.

[Abe02] Masayuki Abe. Securing "Encryption + Proof of Knowledge" in the Random Oracle Model. In Bart Preneel, editor, *Topics in Cryptology - CT-RSA 2002*, volume 2271 of *Lecture Notes in Computer Science*, pages 277–289. Springer, 2002.

[ABN10] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust Encryption. In Daniele Micciancio, editor, *Theory of Cryptography Conference - TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 480–497. Springer, 2010.

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In David Naccache, editor, *Topics in Cryptology - CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158. Springer, 2001.

[AKO09] Masayuki Abe, Eike Kiltz, and Tatsuaki Okamoto. Compact CCA-Secure Encryption for Messages of Arbitrary Length. In Stanislaw Jarecki and Gene Tsudik, editors, *Public Key Cryptography - PKC 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 377–392. Springer, 2009.

[BBDP01] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-Privacy in Public-Key Encryption. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.

[BCP+11] David Bernhard, Véronique Cortier, Olivier Pereira, Ben Smyth, and Bogdan Warinschi. Adapting Helios for Provable Ballot Privacy. In Vijay Atluri and Claudia Díaz, editors, *European Symposium on Research in Computer Security - ESORICS 2011*, volume 6879 of *Lecture Notes in Computer Science*, pages 335–354. Springer, 2011.

[BDPR98] Mihir Bellare, Anand Desai, David Pointcheval, and Phillip Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer, 1998.

[Boy07] Xavier Boyen. Miniature CCA2 PK Encryption: Tight Security Without Redundancy. In Kaoru Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 485–501. Springer, 2007.

[BP04] Mihir Bellare and Adriana Palacio. Towards Plaintext-Aware Public-Key Encryption Without Random Oracles. In Pil Joong Lee, editor, *Advances in Cryptology - ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 48–62. Springer, 2004.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal Asymmetric Encryption. In Alfredo De Santis, editor, *Advances in Cryptology - EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer, 1994.

[BR06] Mihir Bellare and Thomas Ristenpart. Multi-Property-Preserving Hash Domain Extension and the EMD Transform. In Xuejia Lai and Kefei Chen, editors, *Advances in Cryptology - ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 299–314. Springer, 2006.

[CKS08] David Cash, Eike Kiltz, and Victor Shoup. The Twin Diffie-Hellman Problem and Applications. In Nigel P. Smart, editor, *Advances in Cryptology - EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 127–145. Springer, 2008.

[CP92] David Chaum and Torben P. Pedersen. Wallet Databases with Observers. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer, 1992.

[CS98] Ronald Cramer and Victor Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998.

[CS03] Ronald Cramer and Victor 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.

[Dam91] Ivan Damgård. Towards Practical Public Key Systems Secure Against Chosen Ciphertext Attacks. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456. Springer, 1991.

[DY83] Danny Dolev and Andrew Chi-Chih Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–207, 1983.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[Fis00] Marc Fischlin. A Note on Security Proofs in the Generic Model. In Tatsuaki Okamoto, editor, *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 458–469. Springer, 2000.

[Fis05] Marc Fischlin. Communication-Efficient Non-interactive Proofs of Knowledge with Online Extractors. In Victor Shoup, editor, *Advances in Cryptology - CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 152–168. Springer, 2005.

[GJKW07] Eu-Jin Goh, Stanislaw Jarecki, Jonathan Katz, and Nan Wang. Efficient Signature Schemes with Tight Reductions to the Diffie-Hellman Problems. *Journal of Cryptology*, 20(4):493–514, 2007.

[HHK10] Javier Herranz, Dennis Hofheinz, and Eike Kiltz. Some (in)sufficient conditions for secure hybrid encryption. *Inf. Comput.*, 208(11):1243–1257, 2010.

[HLM03] Jonathan Herzog, Moses Liskov, and Silvio Micali. Plaintext Awareness via Key Registration. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 548–564. Springer, 2003.

[Jak98] Markus Jakobsson. A Practical Mix. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 448–461. Springer, 1998.

[KD04] Kaoru Kurosawa and Yvo Desmedt. A New Paradigm of Hybrid Encryption Scheme. In Matthew K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 426–442. Springer, 2004.

[KM04] Kaoru Kurosawa and Toshihiko Matsuo. How to Remove MAC from DHIES. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *Australasian Conference on Information Security and Privacy - ACISP 2004*, volume 3108 of *Lecture Notes in Computer Science*, pages 236–247. Springer, 2004.

[KPSY09] Eike Kiltz, Krzysztof Pietrzak, Martijn Stam, and Moti Yung. A New Randomness Extraction Paradigm for Hybrid Encryption. In Antoine Joux, editor, *Advances in Cryptology - EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 590–609. Springer, 2009.

[KW03] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM Conference on Computer and Communications Security*, pages 155–164. ACM, 2003.

[Lip10] Helger Lipmaa. On the CCA1-Security of Elgamal and Damgård's Elgamal. In Xuejia Lai, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology - Inscrypt 2010*, volume 6584 of *Lecture Notes in Computer Science*, pages 18–35. Springer, 2010.

[Moh10] Payman Mohassel. A Closer Look at Anonymity and Robustness in Encryption Schemes. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 501–518. Springer, 2010.

[OP01] Tatsuaki Okamoto and David Pointcheval. The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In Kwangjo Kim, editor, *Public Key Cryptography - PKC 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 104–118. Springer, 2001.

[Pas03] Rafael Pass. On Deniability in the Common Reference String and Random Oracle Model. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 316–337. Springer, 2003.

[Poe13] Bertram Poettering. Personal Communication, January 2013.

[PS96] David Pointcheval and Jacques Stern. Security Proofs for Signature Schemes. In Ueli M. Maurer, editor, *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398. Springer, 1996.

[PS00] David Pointcheval and Jacques Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 13(3):361–396, 2000.

[RGK06] Mario Di Raimondo, Rosario Gennaro, and Hugo Krawczyk. Deniable authentication and key exchange. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security - CCS 2006*, pages 400–409. ACM, 2006.

[Sak00] Kazue Sako. An Auction Protocol Which Hides Bids of Losers. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography - PKC 2000*, volume 1751 of *Lecture Notes in Computer Science*, pages 422–432. Springer, 2000.

[Sch91] Claus-Peter Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[SG02] Victor Shoup and Rosario Gennaro. Securing Threshold Cryptosystems against Chosen Ciphertext Attack. *Journal of Cryptology*, 15(2):75–96, 2002.

[Sho02] Victor Shoup. OAEP Reconsidered. *Journal of Cryptology*, 15(4):223–249, 2002.

[SJ00] Claus-Peter Schnorr and Markus Jakobsson. Security of Signed ElGamal Encryption. In Tatsuaki Okamoto, editor, *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 73–89. Springer, 2000.

[TY98] Yiannis Tsiounis and Moti Yung. On the Security of ElGamal Based Encryption. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography - PKC '98*, volume 1431 of *Lecture Notes in Computer Science*, pages 117–134. Springer, 1998.

[Wik08] Douglas Wikström. Simplified Submission of Inputs to Protocols. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *Security and Cryptography for Networks - SCN 2008*, volume 5229 of *Lecture Notes in Computer Science*, pages 293–308. Springer, 2008.

A Additional Security Definitions

A.1 IND-ATK-security for a PKE scheme

Let $\mathtt{PKE} = (\mathtt{PKE.Kg}, \mathtt{PKE.Enc}, \mathtt{PKE.Dec})$ be an encryption scheme, and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary. For $\mathrm{atk} \in \{\mathrm{cpa}, \mathrm{cca1}, \mathrm{cca2}\}$ we define the advantage of $\mathcal{A}$ in breaking the indistinguishability of $\mathtt{PKE}$ as:

$$\mathbf{Adv}^{\mathrm{ind-atk}}_{\mathtt{PKE},\mathcal{A}}(k) = \left| \Pr\left[\mathbf{Exp}^{\mathrm{ind-atk}}_{\mathtt{PKE},\mathcal{A}}(k) = 1\right] - \frac{1}{2} \right| \ ,$$

where the experiment is defined as:

$$\begin{array}{l} \textbf{Experiment } \mathbf{Exp}^{\mathrm{ind-atk}}_{\mathtt{PKE},\mathcal{A}}(k)\text{:} \\ (\mathtt{sk}, \mathtt{pk}) \leftarrow \mathtt{PKE.Kg}(1^k) \\ (St, M_0, M_1) \leftarrow \mathcal{A}_1^{\mathcal{O}_1}(\mathtt{pk}) \\ b \leftarrow_{\$} \{0,1\} \\ \psi^* \leftarrow \mathtt{PKE.Enc}(\mathtt{pk}, M_b) \\ b' \leftarrow \mathcal{A}_2^{\mathcal{O}_2}(\mathtt{pk}, \psi^*, St) \\ \text{Return } (b' = b) \end{array}$$

and the oracles $\mathcal{O}_1$ and $\mathcal{O}_2$ are defined as:

atk	$\mathcal{O}_1(\cdot)$	$\mathcal{O}_2(\cdot)$
cpa	$\emptyset$	$\emptyset$
cca1	$\texttt{PKE.Dec}(\texttt{sk}, \cdot)$	$\emptyset$
cca2	$\texttt{PKE.Dec}(\texttt{sk}, \cdot)$	$\texttt{PKE.Dec}(\texttt{sk}, \cdot)$

with the restriction that $\mathcal{A}_2$ cannot query $\mathcal{O}_2$ on the challenge ciphertext ψ^*. PKE is said IND-ATK-secure if the advantage $\mathbf{Adv}^{\text{ind}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k)$ is negligible for all PPT adversaries $\mathcal{A}$.

A.2 ANON-ATK security for a PKE

Let $\texttt{PKE} = (\texttt{PKE.Kg}, \texttt{PKE.Enc}, \texttt{PKE.Dec})$ be an encryption scheme, and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary. For $\text{atk} \in \{\text{cpa}, \text{cca1}, \text{cca2}\}$ we define the advantage of $\mathcal{A}$ in breaking the anonymity of PKE as:

$$\mathbf{Adv}^{\text{anon}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k) = \left| \Pr\left[\mathbf{Exp}^{\text{anon}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k) = 1\right] - \frac{1}{2} \right| \ ,$$

where the experiment is defined as:

Experiment $\mathbf{Exp}^{\text{anon}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k)$:
$(\texttt{sk}_0, \texttt{pk}_0) \leftarrow \texttt{PKE.Kg}(1^k)$, $(\texttt{sk}_1, \texttt{pk}_1) \leftarrow \texttt{PKE.Kg}(1^k)$
$(St, M) \leftarrow \mathcal{A}_1^{\mathcal{O}_{1,0}, \mathcal{O}_{1,1}}(\texttt{pk}_0, \texttt{pk}_1)$
$b \leftarrow_{\$} \{0, 1\}$
$\psi^* \leftarrow \texttt{PKE.Enc}(\texttt{pk}_b, M)$
$b' \leftarrow \mathcal{A}_2^{\mathcal{O}_{2,0}, \mathcal{O}_{2,1}}(\texttt{pk}_0, \texttt{pk}_1, \psi^*, St)$
Return $(b' = b)$

and the oracles $\mathcal{O}_{1,i}$ and $\mathcal{O}_{2,i}$ are defined for $i = 0, 1$ as:

atk	$\mathcal{O}_{1,i}(\cdot)$	$\mathcal{O}_{2,i}(\cdot)$
cpa	$\emptyset$	$\emptyset$
cca1	$\texttt{PKE.Dec}(\texttt{sk}_i, \cdot)$	$\emptyset$
cca2	$\texttt{PKE.Dec}(\texttt{sk}_i, \cdot)$	$\texttt{PKE.Dec}(\texttt{sk}_i, \cdot)$

with the restriction that $\mathcal{A}_2$ cannot query $\mathcal{O}_{2,0}$ nor $\mathcal{O}_{2,1}$ on the challenge ciphertext ψ^*. PKE is said ANON-ATK-secure if the advantage $\mathbf{Adv}^{\text{anon}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k)$ is negligible for all PPT adversaries $\mathcal{A}$.

A.3 SROB-ATK-security for a PKE

Let $\texttt{PKE} = (\texttt{PKE.Kg}, \texttt{PKE.Enc}, \texttt{PKE.Dec})$ be an encryption scheme, and let $\mathcal{A}$ be an adversary. For $\text{atk} \in \{\text{cpa}, \text{cca}\}$ we define the advantage of $\mathcal{A}$ in breaking the strong robustness of PKE as:

$$\mathbf{Adv}^{\text{srob}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k) = \Pr\left[\mathbf{Exp}^{\text{srob}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k) = 1\right] \ ,$$

where the experiment is defined as:

Experiment $\mathbf{Exp}^{\text{srob}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k)$:
$(\texttt{sk}_0, \texttt{pk}_0) \leftarrow \texttt{PKE.Kg}(1^k)$, $(\texttt{sk}_1, \texttt{pk}_1) \leftarrow \texttt{PKE.Kg}(1^k)$
$C \leftarrow \mathcal{A}^{\mathcal{O}_0, \mathcal{O}_1}(\texttt{pk}_0, \texttt{pk}_1)$
$M_0 = \texttt{PKE.Dec}(\texttt{sk}_0, C)$, $M_1 = \texttt{PKE.Dec}(\texttt{sk}_1, C)$
Return $(M_0 \neq \perp) \wedge (M_1 \neq \perp)$

and the oracles $\mathcal{O}_0$ and $\mathcal{O}_1$ are defined as:

atk	$\mathcal{O}_0(\cdot)$	$\mathcal{O}_1(\cdot)$
cpa	$\emptyset$	$\emptyset$
cca	$\texttt{PKE.Dec}(\texttt{sk}_0, \cdot)$	$\texttt{PKE.Dec}(\texttt{sk}_1, \cdot)$

PKE is said SROB-ATK-secure if the advantage $\mathbf{Adv}^{\text{srob}-\text{atk}}_{\texttt{PKE},\mathcal{A}}(k)$ is negligible for all PPT adversaries $\mathcal{A}$.

A.4 Security notions for a DEM

Let $\texttt{DEM} = (\texttt{DEM.Enc}, \texttt{DEM.Dec})$ be a DEM, and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary. The advantage of $\mathcal{A}$ in breaking the ciphertext indistinguishability of DEM is:

$$\mathbf{Adv}^{\text{ind}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k) = \left| \Pr\left[\mathbf{Exp}^{\text{ind}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k) = 1\right] - \frac{1}{2} \right| \ ,$$

where the experiment is defined as:

Experiment $\mathbf{Exp}^{\text{ind}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k)$:
$K \leftarrow_{\$} \texttt{DEM.KeySp}(k)$
$(St, M_0, M_1) \leftarrow \mathcal{A}_1(1^k)$
$b \leftarrow_{\$} \{0,1\}$
$\chi^* \leftarrow \texttt{DEM.Enc}(K, M_b)$
$b' \leftarrow \mathcal{A}_2(1^k, \chi^*, St)$
Return $(b' = b)$

DEM is said IND-OT-secure if the advantage $\mathbf{Adv}^{\text{ind}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k)$ is negligible for all PPT adversaries $\mathcal{A}$.

The advantage of $\mathcal{A}$ in breaking the ciphertext integrity of DEM is:

$$\mathbf{Adv}^{\text{int}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k) = \Pr\left[\mathbf{Exp}^{\text{int}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k) = 1\right] \ ,$$

where the experiment is defined as:

Experiment $\mathbf{Exp}^{\text{int}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k)$:
$K^* \leftarrow_{\$} \texttt{DEM.KeySp}(k)$
$(St, M) \leftarrow \mathcal{A}_1(1^k)$
$\chi^* \leftarrow \texttt{DEM.Enc}(K^*, M)$
$\chi \leftarrow \mathcal{A}_2(1^k, \chi^*, St)$
If $(\chi \neq \chi^*)$ and $(\texttt{DEM.Dec}(K^*, \chi) \neq \perp)$
  Return 1
Return 0

DEM is said INT-OT-secure if the advantage $\mathbf{Adv}^{\text{int}-\text{ot}}_{\texttt{DEM},\mathcal{A}}(k)$ is negligible for all PPT adversaries $\mathcal{A}$.

DEM is said AE-OT-secure if it is both IND-OT-secure and INT-OT-secure.

B Hybrid Variant of the Scheme

We recall the definition of a *data encapsulation mechanism* (DEM) introduced in [CS03] (we will not need the notion of *key encapsulation mechanism* (KEM) to describe the hybrid variant of CPS-EG since it does not exactly fit the KEM/DEM framework).

Definition 3 (DEM). *A* data encapsulation mechanism *(DEM) with key space* $\texttt{DEM.KeySp}(k)$ *is a pair of polynomial-time algorithms* $(\texttt{DEM.Enc}, \texttt{DEM.Dec})$ *where:*

- $\texttt{DEM.Enc}$*, the (deterministic)* data encapsulation algorithm*, takes a key* $K \in \texttt{DEM.KeySp}(k)$ *and a message* M *and returns a ciphertext* χ*.*
- $\texttt{DEM.Dec}$*, the (deterministic)* data decapsulation mechanism*, takes a key* $K \in \texttt{DEM.KeySp}(k)$ *and a ciphertext* χ *and returns either a message* M *or the special symbol* $\perp$ *that indicates that the ciphertext is invalid.*

We will be interested in the following security notion for a DEM [KPSY09]: one-time indistinguishability (IND-OT), which requires that efficient adversaries fail to distinguish the encryption of two messages of their choice. The formal definition is provided in Appendix A.

We now describe the hybrid variant of CPS-EG encryption (HCPS-EG for short), using some DEM $\texttt{DEM} = (\texttt{DEM.Enc}, \texttt{DEM.Dec})$. Let $\boldsymbol{H_c} : \{0,1\}^* \times \mathbb{G}^4 \to \mathbb{Z}_p$ and $\boldsymbol{H_K} : \mathbb{G}^2 \to \cup_{k\in\mathbb{N}} \texttt{DEM.KeySp}(k)$ be two random oracles (we assume that $\boldsymbol{H_K}$ is implicitly given the security parameter and selects the according key space to generate its outputs). The scheme is defined in Figure 4. Note that unlike for the CPS-EG scheme, we use here the optimized variant (c, s) of Schnorr signatures.

HCPS-EG PKE scheme

$\texttt{PKE.Kg}(1^k)$	$\texttt{PKE.Enc}(\texttt{pk} = X, M)$	$\texttt{PKE.Dec}(\texttt{sk} = x, \psi)$
$(\mathbb{G}, p, G) \leftarrow \texttt{GpGen}(1^k)$	$r, a \leftarrow_\$ \mathbb{Z}_p^*$	Parse ψ as (χ, R, c, s)
$x \leftarrow_\$ \mathbb{Z}_p^*$; $X := G^x$	$R := G^r$; $R' := X^r$	$R' := R^x$
$\texttt{sk} := x$; $\texttt{pk} := X$	$K := \boldsymbol{H_K}(R, R')$	$A := G^s R^{-c}$; $A' := A^x$
Return $(\texttt{sk}, \texttt{pk})$	$\chi \leftarrow \texttt{DEM.Enc}(K, M)$	if $\boldsymbol{H_c}(\chi, R, R', A, A') \neq c$
	$A := G^a$; $A' := X^a$	Return $\perp$
	$c := \boldsymbol{H_c}(\chi, R, R', A, A')$	$K := \boldsymbol{H_K}(R, R')$
	$s := a + cr \mod p$	Return $M \leftarrow \texttt{KEM.Dec}(K, \chi)$
	Return $\psi := (\chi, R, c, s)$	

Fig. 4. The HCPS-EG encryption scheme.

Note that this does not fit exactly into the KEM/DEM framework since s and c depend on both the encapsulated key R and χ. The corresponding Schnorr-Signed version was proved IND-CCA2-secure in the ROM + GGM in [SJ00], where it was described with a one-time-pad rather than a general IND-OT-secure DEM. Though we cannot use KEM/DEM composition theorems, we can directly show that the scheme is ROM-PA'-secure (see Section 2.2) and IND-CPA-secure under the CDH assumption and IND-OT-security for the DEM, hence IND-CCA2-secure.

Theorem 10. *The HCPS-EG encryption scheme is ROM-PA'-secure.*

Proof. The proof is quite similar to the proof of Theorem 5, and we only sketch it here. Let $(L_{\boldsymbol{H}}, L_\psi, \psi)$ be the output of a run of a ciphertext creator $\mathcal{C}$, with $\Psi = (\chi, R, c, s)$. The plaintext extractor works as follows: it computes $A = G^s R^{-c}$ and looks throughout $L_{\boldsymbol{H_c}}$ for all queries of the form $(\chi, R, *, A, *)$, where $*$ denotes any group element, such that the answer was c. If there is no such query, it returns $\perp$ (meaning that the ciphertext is invalid). Otherwise, denote (χ, R, R'_i, A, A'_i) all queries of this form whose answer was c. For each i, $\mathcal{P}$ checks whether $X^s = A'_i R'^c_i$. If there is no

such query, the extractor returns $\perp$. If there is more than one such query, the extractor aborts. Otherwise, denoting (χ, R, R', A, A') the unique query such that the check passed, $\mathcal{P}$ makes the query[11] $K := \boldsymbol{H_K}(R, R')$, and returns $M = \mathtt{DEM.Dec}(K, \chi)$. The analysis of the failure probability of the plaintext extractor is exactly the same as for Theorem 5. □

Note that the scheme cannot be proved ROM-PA-secure in general. Indeed, assure that DEM is simply a one-time-pad: $\mathtt{DEM.Enc}(K, M) = K \oplus M$. Consider the following ciphertext creator $\mathcal{C}$: it simply draws a random string χ, and generates a ciphertext $\psi = (\chi, R, c, s)$ by computing the signature "honestly" (*i.e.* by drawing random integers r, a, etc.). Since the ciphertext creator does not make the query $\boldsymbol{H_K}(R, R')$, the plaintext extractor cannot recover the corresponding message $M = \chi \oplus \boldsymbol{H_K}(R, R')$, unless it has free access to $\boldsymbol{H_K}$. However, this seems to be rather an artifact of the ROM than a real problem: if $\boldsymbol{H_K}$ were replaced by a standard key derivation function, the problem would disappear.

Theorem 11. *When* DEM *is IND-OT-secure and under the CDH assumption, the HCPS-EG encryption scheme is IND-CPA-secure (and hence also IND-CCA2-secure).*

Proof. We use a sequence of games. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an IND-CPA adversary against the encryption scheme making at most q_h queries in total to $\boldsymbol{H_K}$ and $\boldsymbol{H_c}$. We start from game Γ_0 which is simply the IND-CPA security experiment defined in Appendix A. Let S_0 be the event that $b' = b$ in game Γ_0. By definition:

$$\mathbf{Adv}^{\mathrm{ind-cpa}}_{\text{HCPS-EG},\mathcal{A}}(k) = \left|\Pr[S_0] - \frac{1}{2}\right| \ .$$

In game Γ_1, we modify the way the challenge ciphertext $\psi^* = (\chi^*, R, c, s)$ is generated. When $\mathcal{A}_1$ outputs the two messages M_0 and M_1, they are forwarded to the challenger of an IND-OT experiment for DEM, which returns some DEM ciphertext χ^* used to build ψ^*. The remaining is unchanged compared with Γ_0. Let S_1 denote the event that $b' = b$ in game Γ_1. It is easy to see that combining $\mathcal{A}$ with the adequate parts of the IND-CPA security experiment against PKE yields an adversary $\mathcal{A}'$ against IND-OT of DEM. Again, by definition:

$$\mathbf{Adv}^{\mathrm{ind-ot}}_{\mathtt{DEM},\mathcal{A}'}(k) = \left|\Pr[S_1] - \frac{1}{2}\right| \ .$$

Moreover, let $\mathtt{Bad}_1$ denote the event that $\mathcal{A}_2$ queries $\boldsymbol{H_K}$ at (R, R') in game Γ_1. Again, it is easy to see that games Γ_0 and Γ_1 are identical until $\mathtt{Bad}_1$ happens, so that:

$$|\Pr[S_1] - \Pr[S_0]| \leq \Pr[\mathtt{Bad}_1] \ .$$

It remains to upper bound $\Pr[\mathtt{Bad}_1]$. For this, we build a reduction $\mathcal{R}$ that solves the CDH problem. Let $(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k)$ and $(X = G^x, R = G^r)$ denote the input to $\mathcal{R}$. $\mathcal{R}$ simulates game Γ_1 as follows. It sets $\mathtt{pk} = X$. All queries of $\mathcal{A}_1$ to $\boldsymbol{H_K}$ or $\boldsymbol{H_c}$ are answered uniformly at random. When $\mathcal{A}_1$ outputs the two messages M_0 and M_1, $\mathcal{R}$ draws a random key $K \leftarrow_{\$} \mathtt{DEM.KeySp}(k)$, a random bit $b \leftarrow_{\$} \{0, 1\}$, and encrypts $\chi^* \leftarrow \mathtt{DEM.Enc}(K, M_b)$. It also draws $c, s \leftarrow_{\$} \mathbb{Z}_p$, and gives $\psi^* = (\chi^*, R, c, s)$, where R is the second element of its CDH challenge, as the challenge ciphertext to $\mathcal{A}_2$. Queries of $\mathcal{A}_2$ to $\boldsymbol{H_K}$ are again answered uniformly at random. Queries of $\mathcal{A}_2$ to $\boldsymbol{H_c}$ are answered as follows in order to ensure the validity of the challenge ciphertext. Denote $A = G^s R^{-c}$. When a query of the form $(\chi^*, R, \tilde{R}, A, \tilde{A})$ is made, the reduction checks whether $X^s = \tilde{A}\tilde{R}^c$. If this holds, the reduction

[11] This is where we need to authorize the plaintext extractor to make additional queries to $\boldsymbol{H}$, in case this query is not included in $L_{\boldsymbol{H_K}}$

answers this query with c. If the check fails, or for any other query not of the form $(\chi^*, R, *, A, *)$, the reduction returns a uniformly random answer. At the end of the simulation, the reduction picks one of the queries $(R, *)$ which were made by $\mathcal{A}_2$ to $\boldsymbol{H_K}$ and returns the second element of this query as its answer to the CDH instance. Assume for a moment that the simulation of game Γ_1 is perfect. Then $\mathcal{R}$ solves the CDH problem with advantage greater than $\Pr[\mathtt{Bad}_1]/q_h$. By inspection, the simulation deviates from perfection only in the simulation of $\boldsymbol{H_c}$. Namely, the simulation fails if the check $X^s = \tilde{A}\tilde{R}^c$ passes and $\tilde{R} \neq R^x$ or $\tilde{A} \neq A^x$. But according to Lemma 1, this can happen with probability at most $1/p$ for each query. Hence, the simulation fails with probability at most q_h/p (the only other event which could make the simulation fail is if $\mathcal{A}_1$ had queried (χ^*, R, R', A, A'), but since $\mathcal{A}_1$'s view is independent of R, separating $\mathcal{A}_1$'s and $\mathcal{A}_2$'s queries shows that this probability can be absorbed in q_h/p). Hence one obtains:

$$\Pr[\mathtt{Bad}_1] \leq q_h \mathbf{Adv}^{\mathrm{cdh}}_{\mathtt{GpGen},\mathcal{R}}(k) + \frac{q_h}{p} \ .$$

Collecting all inequalities yields:

$$\mathbf{Adv}^{\mathrm{ind-cpa}}_{\text{HCPS-EG},\mathcal{A}}(k) \leq q_h \mathbf{Adv}^{\mathrm{cdh}}_{\mathtt{GpGen},\mathcal{R}}(k) + \mathbf{Adv}^{\mathrm{ind-ot}}_{\mathtt{DEM},\mathcal{A}'}(k) + \frac{q_h}{p} \ ,$$

which concludes the proof. □

Regarding anonymity and robustness of HCPS-EG, we have the following result (as for CPS-EG, HCPS-EG* denotes the scheme with the additional check on decryption that $R \neq 1_{\mathbb{G}}$).

Theorem 12. *Assume that the CDH problem is hard for* GpGen*. Then the HCPS-EG encryption scheme is ANON-CCA2-secure (no assumption is needed for the DEM).*
Assume that $\boldsymbol{H_c}$ is instantiated with a collision-resistant hash function family. Then the HCPS-EG* encryption scheme is SROB-CCA-secure.

Proof. The proof for robustness is exactly the same as for Theorem 9. We only sketch the proof for anonymity since it is very similar to the one of Theorems 8 and 11. First, we show ANON-CPA-security of HCPS-EG using a sequence of games. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an ANON-CPA adversary. We let Γ_0 be the original ANON-CPA security game. Denote X_0 and X_1 the public keys given as input to $\mathcal{A}$. Let S_0 be the event that $b' = b$ in game Γ_0. By definition:

$$\mathbf{Adv}^{\mathrm{anon-cpa}}_{\text{HCPS-EG},\mathcal{A}}(k) = \left| \Pr[S_0] - \frac{1}{2} \right| \ .$$

Let Γ_1 be the game similar to Γ_0, except that for generating the challenge ciphertext, the challenger simply draws a random K from $\mathtt{DEM.KeySp}(k)$, a random $r \leftarrow_{\$} \mathbb{Z}_p^*$, random integers $c, s \leftarrow_{\$} \mathbb{Z}_p$, encrypts $\chi = \mathtt{DEM.Enc}(K, M)$, and returns the challenge ciphertext $\Psi^* = (\chi, R, c, s)$ where $R = G^r$. Let S_1 denote the event that $b' = b$ in game Γ_1. Clearly, Ψ^* is independent of b so that $\Pr[S_1] = 1/2$. Moreover, let $\mathtt{Bad}_1$ be the event that $\mathcal{A}$ queries $\boldsymbol{H_K}$ at (R, R') with $R' = X_b^r$ or $\boldsymbol{H_c}$ at (χ, R, R', A, A') with $R' = X_b^r$, $A = G^s R^{-c}$ and $A' = X_b^{s-cr}$. Then games Γ_0 and Γ_1 are identical until $\mathtt{Bad}_1$ happens, so that:

$$|\Pr[S_1] - \Pr[S_0]| \leq \Pr[\mathtt{Bad}_1] \ .$$

One can then conduct the same analysis as in proof of Theorem 11 and show that $\Pr[\mathtt{Bad}_1]$ is negligible when CDH is hard (since intuitively any query provoking $\mathtt{Bad}_1$ will reveal $R' = \mathrm{CDH}(X, R)$). We omit the details.

ANON-CCA2-security follows from a composition theorem similar to the one used for Theorem 8. Namely, one can define the notion of ROM-ANON-PA' security which is similar to the ROM-ANON-PA notion, except that the plaintext extractor is freely allowed to make additional calls to the random oracle. Again, the proof that the scheme satisfies this security notion is very similar to the proof of Theorem 10. □

C Description of Other Related Schemes

We describe here the TDH1/2 encryption schemes, as well as the Twin ElGamal KEM. We stress that TDH1/2 were originally described in [SG02] as hybrid schemes. Here we describe the non-hybrid version. Whereas the hybrid variant of TDH1 is proved IND-CCA2-secure under the CDH assumption, this does not seem to be the case of the non-hybrid variant. Nevertheless, we provide a security proof under the DDH assumption for both schemes. Additionally, we show that TDH1 is not (weakly) robust and that TDH2 is not ANON-CPA-secure.

C.1 The TDH1 encryption scheme

The TDH1 requires an additional random oracle $\boldsymbol{H_G} : \mathbb{G}^3 \to \mathbb{G}$. It is defined in Figure 5.

(non-hybrid) TDH1 PKE scheme

$\mathtt{PKE.Kg}(1^k)$	$\mathtt{PKE.Enc}(\mathtt{pk} = X, M)$	$\mathtt{PKE.Dec}(\mathtt{sk} = x, \psi)$
$(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k)$	$r, a \leftarrow_\$ \mathbb{Z}_p^*$	Parse ψ as $(Y, R, \bar{R}, c, s)$
$x \leftarrow_\$ \mathbb{Z}_p^*$; $X := G^x$	$R := G^r$; $R' := X^r$	$A := G^s R^{-c}$
$\mathtt{sk} := x$; $\mathtt{pk} := X$	$Y := MR'$	$\bar{G} := \boldsymbol{H_G}(Y, R, A)$
Return $(\mathtt{sk}, \mathtt{pk})$	$A := G^a$	$\bar{A} := \bar{G}^s \bar{R}^{-c}$
	$\bar{G} := \boldsymbol{H_G}(Y, R, A)$	if $\boldsymbol{H_c}(\bar{G}, \bar{R}, \bar{A}) \neq c$
	$\bar{R} := \bar{G}^r$; $\bar{A} := \bar{G}^a$	Return $\perp$
	$c := \boldsymbol{H_c}(\bar{G}, \bar{R}, \bar{A})$	$R' := R^x$
	$s := a + cr \mod p$	Return $M := Y/R'$
	Return $\psi := (Y, R, \bar{R}, c, s)$	

Fig. 5. The TDH1 encryption scheme.

Proposition 1. *The TDH1 encryption scheme is IND-CCA2 in the ROM under the DDH assumption.*

Proof. Assume that there exists an IND-CCA2-adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ that can distinguish with non-negligible advantage between the encryption of two messages of its choice. We show that we can build a reduction $\mathcal{R}$ that solves the DDH problem.

Let (X, R^*, R'^*) be the DDH instance given to $\mathcal{R}$. The reduction fixes the public key of the encryption scheme to be $X = G^x$, the secret key being the unknown value x.

Upon reception of a query for $\boldsymbol{H_c}$ or $\boldsymbol{H_G}$, the reduction just checks if a value for the input has already been assigned. If so, it answers with the previously assigned value. Otherwise, it responds with a random value and keeps record of this assignment. For queries to $\boldsymbol{H_G}$, the values are chosen as random powers of X, so that the reduction knows the discrete logarithm of the values in basis X.

The decryption queries (before or after that the adversary received the challenge ciphertext) are treated as follows. Assume the reduction receives a decryption query for a ciphertext $\Psi = (Y, R, \bar{R}, c, s)$. $\mathcal{R}$ first computes $A = G^s R^{-c}$ as usual. If $\boldsymbol{H_G}$ is undefined for input (Y, R, A), then $\mathcal{R}$ returns $\perp$. Otherwise it retrieves the value $\bar{G} = X^t = \boldsymbol{H_G}(Y, R, A)$ (which was effectively set as a random power of X). It also computes $\bar{A} = \bar{G}^s \bar{R}^{-c}$, and can check whether $\boldsymbol{H_c}(\bar{G}, \bar{R}, \bar{A}) = c$. It returns $\perp$ if this does not hold. Otherwise, it computes $R' = \bar{R}^{1/t}$ and returns $M = Y/R'$ as the corresponding plaintext. One can verify that Lemma 1 implies that with overwhelming probability, $\mathtt{DLog}_G(R) = \mathtt{DLog}_{\bar{G}}(\bar{R})$, so that $\bar{R}^{1/t} = X^r$ with $r = \mathtt{DLog}_G(R)$ and the ciphertext is correctly decrypted.

At some point, the adversary sends two messages M_0 and M_1 and waits for the encryption of M_b, $b = 0$ or 1. The reduction then sets $Y^* = M_b R'^*$, with $b \leftarrow_\$ \{0, 1\}$. It also chooses random values $s^*, c^* \leftarrow_\$ \mathbb{Z}_p^*$, and defines $A^* = G^{s^*}(R^*)^{-c^*}$. With overwhelming probability, the query to $\boldsymbol{H_G}$ for the input (Y^*, R^*, A^*) has not been made yet by the adversary. $\mathcal{R}$ sets $\boldsymbol{H_G}(Y^*, R^*, A^*) = X^{t^*}$ for some random t^*. Then it derives $\bar{R}^* = (R'^*)^{t^*}$, $\bar{A}^* = (\bar{G}^*)^{s^*}(\bar{R}'^*)^{-c^*}$ and sets $\boldsymbol{H_c}(\bar{G}^*, \bar{R}^*, \bar{A}^*) = c^*$ (again, with overwhelming probability the adversary did not make the query to $\boldsymbol{H_c}$ for the input $(\bar{G}^*, \bar{R}^*, \bar{A}^*)$). It returns $\Psi^* = (Y^*, R^*, \bar{R}^*, s^*, c^*)$ as the challenge ciphertext. Note that when (X, R^*, R'^*) is a DDH tuple, the encryption of M_b is correctly computed, whereas when (X, R^*, R'^*) is a random tuple, Ψ^* is valid with only negligible probability.

Finally, the adversary will output its guess for b. If the guess is right, the reduction outputs 1. Otherwise it output 0.

Clearly, when (X, R^*, R'^*) is a DDH tuple, the IND-CPA security experiment is perfectly simulated by $\mathcal{R}$ (up to decryption errors which happen only with negligible probability), so that $b' = b$ with probability greater than $1/2 + \mathbf{Adv}^{\mathrm{ind-cpa}}_{\mathtt{PKE},\mathcal{A}}(k)$. When (X, R^*, R'^*) is a random tuple, we would like to argue that the view of $\mathcal{A}$ is independent of b. However, this holds only if $\mathcal{A}_2$ never asks decryption queries $\Psi = (Y, R, \bar{R}, c, s)$ such that $(Y, R, A = G^s R^{-c}) = (Y^*, R^*, A^*)$. We argue now that this can only happen with negligible probability. Assume the contrary, and consider first the case that $(c, s) \neq (s^*, c^*)$. Then $A = A^* = G^s(R^*)^{-c} = G^{s^*}(R^*)^{-c^*}$, which enables the reduction to compute $\mathtt{DLog}_G(R^*)$ and correctly answer the DDH problem. Hence, we may assume that $(c, s) = (s^*, c^*)$, so that necessarily $\bar{R} \neq \bar{R}^*$. This implies however that $\boldsymbol{H_c}(\bar{G}, \bar{R}, \bar{A}) = \boldsymbol{H_c}(\bar{G}, \bar{R}^*, \bar{A}^*) = c$, in other words this yields a collision for $\boldsymbol{H_c}$, which can happen only with negligible probability. Hence when (X, R^*, R'^*) is a random tuple, $\mathcal{R}$ outputs 1 with probability negligibly close to $1/2$, which concludes the proof. □

We conjecture that the TDH1 encryption scheme is ANON-CCA2-secure under the DDH assumption. However, as it is rather a simple observation (captured by the following proposition) that it is not robust, we do not pursue it further.

Proposition 2. *The TDH1 encryption scheme is not (even weakly) robust.*

Proof. The key point is that the validity check is independent of the secret and the public key, so that when a ciphertext $\psi = (Y, R, \bar{R}, c, s)$ is (honestly) created with respect to some public key, then the validity check passes for any other secret key and the ciphertext decrypts to some valid plaintext under any secret key. Hence the scheme cannot be even weakly robust. □

C.2 The TDH2 encryption scheme

The TDH2 scheme is similar to TDH1, except that the element $\bar{G}$ is fixed and included in the public key. It is defined in Figure 6.

(non-hybrid) TDH2 PKE scheme

$\mathtt{PKE.Kg}(1^k)$	$\mathtt{PKE.Enc}(\mathtt{pk} = (X, \bar{G}), M)$	$\mathtt{PKE.Dec}(\mathtt{sk} = x, \psi)$
$(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k)$	$r, a \leftarrow_\$ \mathbb{Z}_p^*$	Parse ψ as $(Y, R, \bar{R}, c, s)$
$x \leftarrow_\$ \mathbb{Z}_p^*$; $X := G^x$	$R := G^r$; $R' := X^r$	$A := G^s R^{-c}$; $\bar{A} := \bar{G}^s \bar{R}^{-c}$
$\bar{G} \leftarrow_\$ \mathbb{G}^*$	$Y := MR'$	if $\boldsymbol{H_c}(Y, R, \bar{R}, A, \bar{A}) \neq c$
$\mathtt{sk} := x$; $\mathtt{pk} := (X, \bar{G})$	$\bar{R} := \bar{G}^r$	Return $\perp$
Return $(\mathtt{sk}, \mathtt{pk})$	$A := G^a$; $\bar{A} := \bar{G}^a$	$R' := R^x$
	$c := \boldsymbol{H_c}(Y, R, \bar{R}, A, \bar{A})$	Return $M := Y/R'$
	$s := a + cr \mod p$	
	Return $\psi := (Y, R, \bar{R}, c, s)$	

Fig. 6. The TDH2 encryption scheme.

Proposition 3. *The TDH2 encryption scheme is IND-CCA2 in the ROM under the DDH assumption.*

Proof. The proof is very similar to the one for TDH1. The only difference is that the reduction fixes the public key of the encryption scheme to be $X = G^x$, and also sets $\bar{G} = X^t$ for some known random value $t \in \mathbb{Z}_p^*$. The simulation of decryption queries can then be done as in the proof of Proposition 1. The rest of the proof is unchanged. □

Proposition 4. *The TDH2 encryption scheme is not ANON-CPA-secure.*

Proof. Let $\mathtt{pk}_0 = (X_0, \bar{G}_0)$ and $\mathtt{pk}_1 = (X_1, \bar{G}_1)$ be the two public keys given to the adversary in the ANON-CPA-security game. The adversary simply sends a random message M to the challenger. It then receives the challenge ciphertext $\Psi^* = (Y, R, \bar{R}, c, s)$. The adversary can now check the validity of the ciphertext against both public keys. Namely it computes $A = G^s R^c$, $\bar{A}_i = \bar{G}_i^s \bar{R}^{-c}$ for $i = 0, 1$, and checks whether $\boldsymbol{H_c}(Y, R, \bar{R}, A, \bar{A}_i) = c$. With overwhelming probability, the validity check will pass for only one public key, therefore revealing the intended recipient. □

Note that if TDH2 is modified so that the same $\bar{G}$ is used for all public keys in the system, then as TDH1, the resulting scheme is not (even weakly) robust.

C.3 The Twin ElGamal KEM

The Twin ElGamal KEM is defined in Figure 7.

Twin-EG KEM

$\mathtt{KEM.Kg}(1^k)$	$\mathtt{KEM.Enc}(\mathtt{pk} = (X_1, X_2))$	$\mathtt{KEM.Dec}(\mathtt{sk} = (x_1, x_2), R)$
$(\mathbb{G}, p, G) \leftarrow \mathtt{GpGen}(1^k)$	$r \leftarrow_\$ \mathbb{Z}_p^*$	$R_1' := R^{x_1}$; $R_2' := R^{x_2}$
$x_1, x_2 \leftarrow_\$ \mathbb{Z}_p^*$	$R := G^r$	$K := \boldsymbol{H_K}(R, R_1', R_2')$
$X_1 := G^{x_1}$; $X_2 := G^{x_2}$	$R_1' := X_1^r$; $R_2' := X_2^r$	Return K
$\mathtt{sk} := (x_1, x_2)$	$K := \boldsymbol{H_K}(R, R_1', R_2')$	
$\mathtt{pk} := (X_1, X_2)$	Return (K, R)	
Return $(\mathtt{sk}, \mathtt{pk})$		

Fig. 7. The Twin ElGamal KEM.