An extended abstract of this paper appears in *Advances in Cryptology – EUROCRYPT '04*, Lecture Notes in Computer Science Vol. , C. Cachin and J. Camenisch ed., Springer-Verlag, 2004. This is the full version.

An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem

Mihir Bellare* Alexandra Boldyreva † Adriana Palacio ‡

Abstract

We present a simple, natural random-oracle (RO) model scheme, for a practical goal, that is uninstantiable, meaning is proven in the RO model to meet its goal yet admits *no* standard-model instantiation that meets this goal. The goal in question is *IND-CCA-preserving asymmetric encryption* which formally captures security of the most common practical usage of asymmetric encryption, namely to transport a symmetric key in such a way that symmetric encryption under the latter remains secure. The scheme is an ElGamal variant, called Hash ElGamal, that resembles numerous existing RO-model schemes, and on the surface shows no evidence of its anomalous properties.

More generally, we show that a certain goal, that we call key-verifiable, ciphertext-verifiable IND-CCA-preserving asymmetric encryption, is achievable in the RO model (by Hash ElGamal in particular) but unachievable in the standard model. This helps us better understand the source of the anomalies in Hash ElGamal and also lifts our uninstantiability result from being about a specific scheme to being about a primitive or goal.

These results extend our understanding of the gap between the standard and RO models, and bring concerns raised by previous work closer to practice by indicating that the problem of RO-model schemes admitting no secure instantiation can arise in domains where RO schemes are commonly designed.

*Dept. of Computer Science & Engineering, University of California at San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. E-Mail: mihir@cs.ucsd.edu. URL: http://www.cse.ucsd.edu/users/mihir. Supported in part by NSF grant CCR-0098123, NSF grant ANR-0129617 and an IBM Faculty Partnership Development Award.

†Dept. of Computer Science & Engineering, University of California at San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. E-Mail: aboldyre@cs.ucsd.edu. URL: http://www.cse.ucsd.edu/users/aboldyre. Supported in part by above-mentioned grants of first author.

‡Dept. of Computer Science & Engineering, University of California at San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. E-Mail: apalacio@cs.ucsd.edu. URL: http://www.cse.ucsd.edu/users/apalacio. Supported by a National Science Foundation Graduate Research Fellowship.

Contents

1 Introduction

A random-oracle (RO) model scheme is one whose algorithms have oracle access to a random function. Its security is evaluated with respect to an adversary with oracle access to the same function. An "instantiation" of such a scheme is the standard-model scheme obtained by replacing this function with a member of a polynomial-time computable family of functions, described by a short key. The security of the scheme is evaluated with respect to an adversary given the same key. In the random-oracle paradigm, as enunciated by Bellare and Rogaway [5], one first designs and proves secure a scheme in the RO model, and then instantiates it to get a (hopefully still secure) standard-model scheme.

The RO model has proven quite popular and there are now numerous practical schemes designed and proven secure in this model. But the important issue of whether such schemes can be securely instantiated, and, if so, how, remains less clear. This paper adds to existing concerns in this regard. Let us begin by reviewing previous work and then explain our results.

1.1 Previous work

Let us call a RO-model scheme *uninstantiable*, with respect to some underlying cryptographic goal, if the scheme can be proven to meet this goal in the random-oracle model, but *no* instantiation of the scheme meets the goal in question.

Canetti, Goldreich and Halevi [7] provided the first examples of uninstantiable schemes, the goals in question being IND-CPA-secure asymmetric encryption and digital signatures secure against chosen-message attacks. Further examples followed: Nielsen [18] presented an uninstantiable RO-model scheme for the goal of non-interactive, non-committing encryption [6], and Goldwasser and Taumann [16] showed the existence of a 3-move protocol which, when collapsed via a RO as per the Fiat-Shamir heuristic [13], yields an uninstantiable RO-model signature scheme.

The results of [7] indicate that it is possible for the RO paradigm to fail to yield secure "real-world" schemes. The example schemes provided by [7], however, are complex and contrived ones that do not resemble the kinds of RO schemes typically being designed. (Their schemes are designed to return the secret key depending on the result of some test applied to an output of the oracle, and they use diagonalization and CS proofs [17].) The same is true of the scheme of [16]. In contrast, the scheme of [18] is simple, but the goal, namely non-interactive, non-committing encryption, is somewhat distant from ones that are common practical targets of RO-model designs. Accordingly, based on existing work, one might be tempted to think that "in practice," or when confined to "natural" schemes for practical problems commonly being targeted by RO-scheme designers, the RO paradigm is sound.

This paper suggests that even this might not always be true. For a practical cryptographic goal, we present an uninstantiable RO-model scheme that is simple and natural, closely resembling the types of schemes being designed in this domain. We begin below by discussing the goal, which we call IND-CCA-preserving asymmetric encryption and which arises in the domain of hybrid encryption.

1.2 IND-CCA-preserving asymmetric encryption

In practice, the most common usage of asymmetric encryption is to transport a symmetric key that is later used for symmetric encryption of the actual data. The notion of an asymmetric encryption scheme AS being IND-CCA-preserving, that we introduce, captures the security attribute that AS must possess in order to render this usage of AS secure. We now elaborate.

Encryption, in practice, largely employs the "hybrid" paradigm. The version of this paradigm that we consider here is quite general. In a first phase, the sender picks at random a "session" key K for a symmetric encryption scheme, encrypts K asymmetrically under the receiver's public key to get a ciphertext C_a, and transfers C_a to the receiver. In a second phase, it can encrypt messages of its choice symmetrically under K and transfer the corresponding ciphertexts to the receiver. We call this multi-message (mm) hybrid encryption.[1]

A choice of an asymmetric encryption scheme AS and a symmetric encryption scheme SS gives rise to a particular mm-hybrid scheme. We introduce in Section 2 a definition of the IND-CCA security of this mm-hybrid scheme which captures the privacy of the encrypted messages even in the presence of an adversary allowed chosen-ciphertext attacks on both component schemes and allowed to choose the messages to be encrypted adaptively and as a function of the asymmetric ciphertext, denoted C_a above, that transports the symmetric key.

Now let us say that an asymmetric encryption scheme AS is *IND-CCA preserving* if the mm-hybrid associated to AS and symmetric encryption scheme SS is IND-CCA secure for *every* IND-CCA secure SS. This notion of security for an asymmetric encryption scheme captures the security attribute of its being able to securely transport a session key for the purpose of mm-hybrid encryption. The goal we consider is IND-CCA-preserving asymmetric encryption.

It is easy to see that any IND-CCA-secure asymmetric encryption scheme is IND-CCA preserving. (For completeness, this is proved in Appendix B.1.) IND-CCA preservation, however, is actually a weaker requirement on an asymmetric encryption scheme than IND-CCA security itself. In fact, since the messages to be encrypted using the asymmetric scheme are randomly-chosen symmetric keys, the encryption itself need not even be randomized. Hence there might be IND-CCA-preserving asymmetric encryption schemes that are simpler and more efficient than IND-CCA-secure ones. In particular, it is natural to seek an efficient IND-CCA-preserving scheme in the RO model along the lines of existing hybrid encryption schemes such as those of [8, 9, 14, 19].

1.3 The Hash ElGamal scheme and its security

It is easy to see that the ElGamal encryption scheme [12] is not IND-CCA preserving. An effort to strengthen it to be IND-CCA preserving lead us to a variant that we call the Hash ElGamal scheme. It uses the idea underlying the Fujisaki-Okamoto [14] transformation, namely to encrypt under the original (ElGamal) scheme using coins obtained by applying a random oracle H to the message. Specifically, encryption of a message K under public key (q, g, X) in the Hash ElGamal scheme is given by

$$\mathsf{AE}^{G,H}((q,g,X),K) \;=\; (g^{H(K)}\,,\,G(X^{H(K)}) \oplus K)\;, \tag{1}$$

where G, H are random oracles, $q, 2q+1$ are primes, g is a generator of the order q cyclic subgroup of $\mathbb{Z}^*_{2q+1}$, and the secret key is (q, g, x) where $g^x = X$. Decryption is performed in the natural way as detailed in Figure 1.

The Hash ElGamal scheme is very much like practical RO-model schemes presented in the literature. In fact, it is a particular case of an asymmetric encryption scheme proposed by Baek, Lee and Kim [2, 3].

We note that the Hash ElGamal asymmetric encryption scheme is not IND-CCA secure, or even

[1] The term multi-message refers to the fact that multiple messages may be encrypted, in the second phase, under the same session key. The main reason for using such a hybrid paradigm, as opposed to directly encrypting the data asymmetrically under the receiver's public key, is that the number-theoretic operations underlying popular asymmetric encryption schemes are computationally more expensive than the block-cipher operations underlying symmetric encryption schemes, so hybrid encryption brings significant performance gains.

IND-CPA secure, in particular because the encryption algorithm is deterministic. But Theorem 3.1 guarantees that the Hash ElGamal asymmetric encryption scheme is IND-CCA-preserving in the RO model, if the Computational Diffie-Hellman (CDH) problem is hard in the underlying group.

We follow this with Theorem 4.1, however, which says that the Hash ElGamal scheme is uninstantiable. In other words, the standard-model asymmetric encryption scheme obtained by instantiating the RO-model Hash ElGamal scheme is not IND-CCA preserving, regardless of the choice of instantiating functions.[2] (We allow these to be drawn from any family of polynomial-time computable functions.)

1.4 A closer look

As noted above, we show that no instantiation of the Hash ElGamal scheme is IND-CCA-preserving. The way we establish this is the following. We let AS be some (any) instantiation of the Hash ElGamal scheme. Then, we construct a particular IND-CCA-secure symmetric encryption scheme SS such that the mm-hybrid associated to AS and SS is not IND-CCA secure. The latter is proven by presenting an explicit attack on the mm-hybrid. We clarify that the symmetric scheme SS constructed in this proof is not a natural one. It is contrived, but not particularly complex. We do not view this as subtracting much from the value of our result, which lies rather in the nature of the Hash ElGamal scheme itself and the practicality of the underlying goal.

What we suggest is interesting about the result is that the Hash ElGamal scheme, on the surface, seems innocuous enough. It does not seem to be making any "peculiar" use of its random oracle that would lead us to think it is "wrong." (Indeed, it uses random oracles in ways they have been used previously, in particular by [14, 2, 3].) The scheme is simple, efficient, and similar to other RO-model schemes out there. In addition, we contend that the definition of IND-CCA-preserving asymmetric encryption is natural and captures a practical requirement. The fact that the Hash ElGamal scheme is uninstantiable thus points to the difficulty of being able to distinguish uninstantiable RO-model schemes from ones that at least *may* be securely instantiable, even in the context of natural and practical goals.

1.5 Generalizations

Next we provide some results that generalize the above. We consider the class of IND-CCA-preserving asymmetric encryption schemes that possess a pair of properties that we call *key verifiability* and *ciphertext verifiability*. Key verifiability means there is a way to recognize valid public keys in polynomial time. Ciphertext verifiability means there is a polynomial-time procedure to determine whether a given ciphertext is an encryption of a given message under a given valid public key. Note that ciphertext verifiability contradicts IND-CPA security, but it need not prevent a scheme from being IND-CCA preserving, since the latter notion considers the use of the asymmetric scheme only for the encryption of messages that are chosen at random.

Theorem 5.2 points out that the goal of key-verifiable, ciphertext-verifiable IND-CCA-preserving asymmetric encryption is achievable in the RO model, by the Hash El Gamal scheme in particular, assuming the CDH problem is hard in the underlying group. Theorem 5.3, however, says that this goal is not achievable in the standard model. In other words, there exist RO-model schemes meeting this goal, but there exist no standard-model schemes meeting it. Theorem 5.3

[2] This result is based on the assumption that one-way functions exist (equivalently, IND-CCA-secure symmetric encryption schemes exist), since, otherwise, by default, *any* asymmetric encryption scheme is IND-CCA preserving, and, indeed, the entire mm-hybrid encryption problem we are considering is vacuous. This assumption is made implicitly in all results in this paper.

generalizes Theorem 4.1 because any instantiation of the Hash ElGamal scheme is key-verifiable and ciphertext-verifiable, and hence cannot be IND-CCA-preserving.

Theorem 5.3 lifts our results from being about a particular scheme to being about a primitive, or class of schemes. The generalization also helps better understand what aspects of the Hash ElGamal scheme lead to its admitting no IND-CCA-preserving instantiation. In particular, we see that this is not due to some "peculiar" use of random oracles but rather due to some simply stated properties of the resulting asymmetric encryption scheme itself.

1.6 Related work

In the cryptographic community, the term "hybrid encryption" seems to be used quite broadly, to refer to a variety of goals or methods in which symmetric and asymmetric primitives are combined to achieve privacy. We have considered one goal in this domain, namely mm-hybrid encryption. We now discuss related work that has considered other goals or problems in this domain.

Works such as [8, 9, 14, 19, 11, 20] provide designs of IND-CCA-secure asymmetric encryption schemes that are referred to as "hybrid encryption schemes" because they combine the use of asymmetric and symmetric primitives. (Possible goals of such designs include gaining efficiency, increasing the size of the message space, or reducing the assumptions that must be made on the asymmetric component in order to guarantee the IND-CCA security of the construction.) The schemes of [8, 9, 14, 19] are in the RO model and, although addressing a different goal, form an important backdrop for our work because the Hash ElGamal scheme is based on similar techniques and usage of random oracles. We stress, however, that we have no reason to believe that any of these schemes, or that of [2, 3] of which Hash ElGamal is a special case, are uninstantiable.

2 Definitions

Notation and conventions. If S is a randomized algorithm, then $[S(x, y, \ldots)]$ denotes the set of all points having positive probability of being output by S on inputs $x, y, \ldots$. If x is a binary string, then $|x|$ denotes its length, and if $n \geq 1$ is an integer, then $|n|$ denotes the length of its binary encoding, meaning the unique integer ℓ such that $2^{\ell-1} \leq n < 2^{\ell}$. The string-concatenation operator is denoted "$\|$".

Formal definitions in the RO model provide as an oracle, to the algorithms and the adversary, a single random function R mapping $\{0,1\}^*$ to $\{0,1\}$. Schemes might, however, use and refer to multiple random functions of different domains and ranges. These can be derived from R via standard means [5].

Symmetric encryption. A symmetric encryption scheme $\mathsf{SS} = (\mathsf{SK}, \mathsf{SE}, \mathsf{SD})$ is specified by three polynomial-time algorithms: via $K \stackrel{\$}{\leftarrow} \mathsf{SK}(1^k)$ one can generate a key; via $C \stackrel{\$}{\leftarrow} \mathsf{SE}(K, M)$ one can encrypt a message $M \in \{0,1\}^*$; and via $M \leftarrow \mathsf{SD}(K, C)$ one can decrypt a ciphertext C. It is required that $\mathsf{SD}(K, \mathsf{SE}(K, M)) = M$ for all $K \in [\mathsf{SK}(1^k)]$ and all $M \in \{0,1\}^*$. We assume (without loss of generality) that $[\mathsf{SK}(1^k)] \subseteq \{0,1\}^k$. In the RO model, all algorithms have access to the RO.

We define security following [4] and addressing the possibility of the symmetric scheme being in the RO model. Let $\mathrm{LR}(M_0, M_1, b) = M_b$ if M_0, M_1 are strings of equal length, and $\perp$ otherwise. Associate to SS, an adversary $\boldsymbol{S}$, and $k \in \mathbb{N}$, the following experiment.

Experiment $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k)$

Randomly choose RO $R_s\colon \{0,1\}^* \to \{0,1\}$

$K \stackrel{\$}{\leftarrow} \mathsf{SK}^{R_s}(1^k)$; $b \stackrel{\$}{\leftarrow} \{0,1\}$

Run $\boldsymbol{S}$ with input 1^k and oracles $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{SD}^{R_s}(K,\cdot)$, R_s
Let d denote the output of $\boldsymbol{S}$
If $d = b$ then return 1 else return 0.

We say that adversary $\boldsymbol{S}$ is legitimate if it never queries $\mathsf{SD}^{R_s}(K,\cdot)$ with a ciphertext previously returned by $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$. Symmetric encryption scheme SS is said to be IND-CCA secure if the function

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) \;=\; 2\cdot\Pr\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) = 1\,\right] - 1$$

is negligible for all legitimate polynomial-time adversaries $\boldsymbol{S}$.

Asymmetric encryption. An asymmetric encryption scheme $\mathsf{AS} = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ is specified by three polynomial-time algorithms: via $(pk, sk) \stackrel{\$}{\leftarrow} \mathsf{AK}(1^k)$ one can generate keys; via $C \stackrel{\$}{\leftarrow} \mathsf{AE}(pk, K)$ one can encrypt a message $K \in \{0,1\}^k$; and via $K \leftarrow \mathsf{AD}(sk, C)$ one can decrypt a ciphertext C. (We denote the message by K because we will set it to a key for a symmetric encryption scheme.) It is required that $\mathsf{AD}(sk, \mathsf{AE}(pk, K)) = K$ for all $(pk, sk) \in [\mathsf{AK}(1^k)]$ and all $K \in \{0,1\}^k$. In the RO model, all algorithms have access to the RO.

Discussions and peripheral results in this paper sometimes refer to standard notions of security for such schemes like IND-CPA and IND-CCA, but these are not required for the main results and, accordingly, are not defined here but recalled in Appendix B.1.

IND-CCA-preserving asymmetric encryption. We provide the formal definitions first and explanations later. A *multi-message hybrid (mm-hybrid) encryption scheme* is simply a pair $(\mathsf{AS}, \mathsf{SS})$ consisting of an asymmetric encryption scheme $\mathsf{AS} = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ and a symmetric encryption scheme $\mathsf{SS} = (\mathsf{SK}, \mathsf{SE}, \mathsf{SD})$. We associate to $(\mathsf{AS}, \mathsf{SS})$, a *hybrid adversary* $\boldsymbol{H}$, and $k \in \mathbb{N}$, the following experiment.

Experiment $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k)$
Randomly choose RO R: $\{0,1\}^* \to \{0,1\}$
Define ROs $R_s(\cdot) = R(0\|\cdot)$ and $R_a(\cdot) = R(1\|\cdot)$
$(pk, sk) \stackrel{\$}{\leftarrow} \mathsf{AK}^{R_a}(1^k)$; $K \stackrel{\$}{\leftarrow} \mathsf{SK}^{R_s}(1^k)$; $b \stackrel{\$}{\leftarrow} \{0,1\}$
$C_a \stackrel{\$}{\leftarrow} \mathsf{AE}^{R_a}(pk, K)$
Run $\boldsymbol{H}$ with inputs pk, C_a and oracles $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{SD}^{R_s}(K,\cdot)$, $\mathsf{AD}^{R_a}(sk,\cdot)$, R
Let d denote the output of $\boldsymbol{H}$
If $d = b$ then return 1 else return 0.

We say that adversary $\boldsymbol{H}$ is legitimate if it does not query $\mathsf{SD}^{R_s}(K,\cdot)$ on a ciphertext previously returned by $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$, and it does not query $\mathsf{AD}^{R_a}(sk,\cdot)$ on C_a. Mm-hybrid encryption scheme $(\mathsf{AS}, \mathsf{SS})$ is said to be IND-CCA secure if the function

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) \;=\; 2\cdot\Pr\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) = 1\,\right] - 1$$

is negligible for all legitimate polynomial-time adversaries $\boldsymbol{H}$.

Finally, we say that an asymmetric encryption scheme AS is *IND-CCA preserving* if the mm-hybrid encryption scheme $(\mathsf{AS}, \mathsf{SS})$ is IND-CCA secure for *all* IND-CCA-secure symmetric encryption schemes SS. Here, the set of symmetric encryption schemes over which we quantify includes RO-model ones if AS is a RO-model scheme, and includes only standard-model ones if AS is a standard-model scheme.

Let us now explain the ideas behind these formalisms. Recall that we are modelling the security of the following two-phase scenario: in phase one, the sender picks a key K for symmetric encryption, asymmetrically encrypts it under the receiver's public key to get a ciphertext C_a, and sends C_a to the receiver; in phase two, the sender symmetrically encrypts messages of its choice under K and transmits the resulting ciphertexts to the receiver. The definition above captures the requirement of privacy of the symmetrically encrypted data under a chosen-ciphertext attack. Privacy is formalized in terms of indistinguishability via left-or-right oracles, and the chosen-ciphertext attack is formalized via the adversary's access to decryption oracles for *both* the symmetric and asymmetric schemes. The legitimacy requirement, as usual, disallows decryption queries on challenge ciphertexts since they would lead to trivial adversary victory. The experiment reflects the possibility that SS and AS are RO-model schemes by picking random oracles for their encryption and decryption algorithms. The standard model is the special case where the algorithms of the schemes do not refer to any oracles, and thus the definition above covers security in both models. The notion of AS being IND-CCA preserving reflects a valuable pragmatic requirement, namely that one may use, in conjunction with AS, any symmetric encryption scheme and be guaranteed security of the mm-hybrid under the minimal assumption that the symmetric scheme itself is secure.

Remark 2.1 Suppose we have two RO-model schemes, and are composing them, or executing them in a common context. (Above, this is happening with the asymmetric encryption scheme and the symmetric encryption scheme.) We claim that, in this case, the ROs of the two schemes should be chosen independently. (This does not mean that we need to assume two RO oracles are given. The formal model always provides just one RO. But one can easily derive several independent ROs from a single one, as we did above.) The correctness of this principle of independent instantiation of ROs in a common context can be seen in many ways. First, it is easy to come up with an example of a pair of secure RO-model schemes that, when composed, yield an insecure one if the ROs in the two schemes are defined to be the same. Second, one can reason by analogy with the way we need to choose keys in composing primitives. For example, suppose we have a MAC and symmetric encryption scheme, each individually secure. If we use them to construct an authenticated-encryption scheme, we should use different keys for the MAC and the symmetric encryption scheme. (There is no reason to think otherwise that the composition will be secure.) The principle, for ROs, is exactly the same. They are just like keys provided to primitives. ∎

The existence of IND-CCA-preserving asymmetric encryption schemes is easy to establish since, as Theorem B.1 indicates, any IND-CCA-secure asymmetric encryption scheme is IND-CCA preserving. The interesting question is to find IND-CCA-preserving asymmetric encryption schemes that are more efficient than existing IND-CCA-secure asymmetric encryption schemes. Hash El Gamal is one such scheme.

3 The HEG scheme and its security in the RO model

In this section we introduce a variant of the ElGamal encryption scheme [12] that, although not IND-CCA secure, is IND-CCA preserving in the RO model under a standard assumption. In Section 4, we will show that this scheme admits no IND-CCA-preserving instantiation.

Preliminaries. A *cyclic-group generator* is a randomized, polynomial-time algorithm CG which on input 1^k outputs a pair (q, g), where q is a prime such that $p = 2q + 1$ is also a prime, g is a generator of the cyclic, order q subgroup $\langle g \rangle$ of $\mathbb{Z}_p^*$, and $|p| = k$. Recall that the Computational Diffie-Hellman (CDH) problem is said to be hard for CG if the function

$$\mathsf{Adv}^{\mathrm{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) \;=\; \Pr\left[\,(q,g) \stackrel{\$}{\leftarrow} \mathsf{CG}(1^k)\,;\, x,y \stackrel{\$}{\leftarrow} \mathbb{Z}_q \;:\; \boldsymbol{C}(q,g,g^x,g^y) = g^{xy}\,\right]$$

is negligible for all polynomial-time *cdh adversaries* $\boldsymbol{C}$.

$\mathsf{AK}(1^k)$	$\mathsf{AE}^{G,H}((q,g,X),K)$	$\mathsf{AD}^{G,H}((q,g,x),(Y,W))$
$(q,g) \xleftarrow{\$} \mathsf{CG}(1^k)$ $x \xleftarrow{\$} \mathbb{Z}_q$ $X \leftarrow g^x$ Return $((q,g,X),(q,g,x))$	$y \leftarrow H(K)$ $Y \leftarrow g^y$ $T \leftarrow G(X^y)$ $W \leftarrow T \oplus K$ Return (Y,W)	$T \leftarrow G(Y^x)$ $K \leftarrow T \oplus W$ If $g^{H(K)} = Y$ then Return K else Return $\perp$ EndIf

Figure 1: Algorithms of the RO-model asymmetric encryption scheme $\mathsf{HEG}[\mathsf{CG}] = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ associated to cyclic-group generator CG. Here $G\colon \langle g \rangle \to \{0,1\}^k$ and $H\colon \{0,1\}^k \to \mathbb{Z}_q$ are random oracles.

3.1 Scheme and result statement

To any cyclic-group generator CG we associate the RO-model asymmetric encryption scheme $\mathsf{HEG}[\mathsf{CG}] = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ whose constituent algorithms are depicted in Figure 1. (The scheme makes reference to two ROs, namely $G\colon \langle g \rangle \to \{0,1\}^k$ and $H\colon \{0,1\}^k \to \mathbb{Z}_q$, while the formal definition of an asymmetric encryption scheme provides a single RO $R\colon \{0,1\}^* \to \{0,1\}$, but G, H may be implemented via R in standard ways [5].) We call this variant of the ElGamal encryption scheme the *Hash ElGamal* encryption scheme associated to CG. Our result about its security in the RO model is the following.

Theorem 3.1 If the CDH problem is hard for cyclic-group generator CG, then the associated Hash ElGamal asymmetric encryption scheme $\mathsf{HEG}[\mathsf{CG}]$ is IND-CCA preserving in the RO model. ▮

For the definition of what it means to be IND-CCA preserving, we refer the reader to Section 2.

Remarks. We note that the encryption algorithm AE of $\mathsf{HEG}[\mathsf{CG}]$ is deterministic. For this reason alone, $\mathsf{HEG}[\mathsf{CG}]$ is not an IND-CCA secure, or even IND-CPA secure, asymmetric encryption scheme. Nonetheless, Theorem 3.1 says that it is IND-CCA preserving as long as the CDH problem is hard for CG. This is not a contradiction. Very roughly, the reason $\mathsf{HEG}[\mathsf{CG}]$ can preserve IND-CCA while not itself being even IND-CPA is that the former notion considers the use of the scheme only for the encryption of messages that are symmetric keys, which (as long as the associated symmetric encryption scheme is secure) have relatively high entropy, and the entropy in these messages compensates for the lack of any introduced by AE. We add that previous work [8, 9, 14, 19] has shown that in the RO model, relatively weak asymmetric components suffice to ensure strong security properties of the hybrid based on them. Thus, it is not surprising that, although $\mathsf{HEG}[\mathsf{CG}]$ is not secure with respect to standard measures like IND-CPA and IND-CCA, it is secure enough to permit its use for transport of a symmetric encryption key as indicated by Theorem 3.1.

3.2 Proof overview

The full proof of Theorem 3.1 is in Appendix A. Here we provide an overview that highlights the main areas of novelty.

Proof setup. Let $\mathsf{AS} = \mathsf{HEG}[\mathsf{CG}]$ and let $\mathsf{AK}, \mathsf{AE}, \mathsf{AD}$ denote its constituent algorithms. Let $\mathsf{SS} = (\mathsf{SK}, \mathsf{SE}, \mathsf{SD})$ be any IND-CCA-secure symmetric encryption scheme. We need to show that $(\mathsf{AS}, \mathsf{SS})$ is an IND-CCA-secure mm-hybrid encryption scheme.

Let $\boldsymbol{H}$ be a polynomial-time hybrid adversary attacking $(\mathsf{AS},\mathsf{SS})$. We will construct polynomial-time adversaries $\boldsymbol{S}$ and $\boldsymbol{C}$ such that

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) \;\le\; \text{poly}(k)\cdot\text{poly}\left(\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k)\,,\,\mathsf{Adv}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k)\right)+\frac{\text{poly}(k)}{2^k}\;. \tag{2}$$

Since SS is assumed IND-CCA secure and the CDH problem is hard for CG, the advantage functions related to $\boldsymbol{S}$ and $\boldsymbol{C}$ above are negligible, and thus so is the advantage function related to $\boldsymbol{H}$. To complete the proof, we need to specify adversaries $\boldsymbol{S},\boldsymbol{C}$ for which Equation (2) is true.

Consider $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k)$. Let (q,g,X) be the public key and (q,g,x) the secret key chosen, where $X=g^x$. Let $C_a=(Y,W)$ where $Y=g^y$. Let K denote the symmetric encryption key chosen. Let GH be the event that there is a time at which g^{xy} is queried to G but K has not been queried to H; HG the event that there is a time at which K is queried to H but g^{xy} has not been queried to G; and $\mathsf{Succ}(\boldsymbol{H})$ the event that $\boldsymbol{H}$ is successful at guessing the value of its challenge bit b. We will construct $\boldsymbol{C}$ so that

$$\Pr\left[\,\mathsf{GH}\,\right] \;\le\; \text{poly}(k)\cdot\mathsf{Adv}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k)+\frac{\text{poly}(k)}{2^k}\;,$$

and we will construct $\boldsymbol{S}$ so that

$$\Pr\left[\,\mathsf{HG}\vee(\mathsf{Succ}(\boldsymbol{H})\wedge\neg\mathsf{GH}\wedge\neg\mathsf{HG})\,\right] \;\le\; \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k)+\frac{\text{poly}(k)}{2^k}\;. \tag{3}$$

Equation (2) follows.

THE ADVERSARIES. The design of $\boldsymbol{C}$ relies mostly on standard techniques, and so we leave it to Appendix A. We turn to $\boldsymbol{S}$. The latter gets input 1^k and oracles $\mathsf{SE}^{R_s}(K,\mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{SD}^{R_s}(K,\cdot)$, R_s, and begins with the initializations

$$((q,g,X),(q,g,x)) \xleftarrow{\$} \mathsf{AK}(1^k)\;;\; y\xleftarrow{\$}\mathbb{Z}_q\;;\; Y\leftarrow g^y\;;\; W\xleftarrow{\$}\{0,1\}^k\;;\; C_a\leftarrow(Y,W)\;. \tag{4}$$

It then runs $\boldsymbol{H}$ on inputs $(q,g,X),C_a$, itself responding to the oracle queries of the latter. Its aim is to do this in such a way that the key K underlying $\boldsymbol{S}$'s oracles plays the role of the quantity of the same name for $\boldsymbol{H}$. Eventually, it will output what $\boldsymbol{H}$ outputs. The difficulty faced by this adversary is that $\boldsymbol{H}$ might query K to H. (Other oracle queries are dealt with in standard ways.) In that case, $\boldsymbol{H}$ expects to be returned y. (And it cannot be fooled since, knowing $Y=g^y$, it can verify whether or not the value returned is y.) The difficulty for $\boldsymbol{S}$ is not that it does not know the right answer —via Equation (4), it actually knows y— but rather that it is not clear how it would know that a query being made to H equals the key K underlying its oracles, so that it would know *when* to return y as the answer to a query to H.

In order to "detect" when query K is made, we would, ideally, like a test that can be performed on a value L, accepting if $L=K$ and rejecting otherwise. However, it is not hard to see that, in general, such a test does not exist.[3] Instead, we introduce a test that has a weaker property and show that it suffices for us.

Our test $\mathsf{KeyTest}$ takes input L and has access to $\boldsymbol{S}$'s $\mathsf{SE}^{R_s}(K,\mathrm{LR}(\cdot,\cdot,b))$ oracle. It returns a pair $(\mathrm{dec},\mathrm{gs})$ such that: (1) If $L=K$ then $(\mathrm{dec},\mathrm{gs})=(1,b)$, meaning in this case it correctly computes the challenge bit b, and (2) If $L\neq K$ then, with overwhelming probability, either $\mathrm{dec}=0$ (the test is saying $L\neq K$) or $(\mathrm{dec},\mathrm{gs})=(1,b)$ (the test is saying it does not know whether or not $L=K$, but it has successfully calculated the challenge bit anyway). With $\mathsf{KeyTest}$ in hand, $\boldsymbol{S}$ can answer a query L made to H as follows. It runs $(\mathrm{dec},\mathrm{gs})\xleftarrow{\$}\mathsf{KeyTest}(L)$. If $\mathrm{dec}=0$, it can safely assume

[3] Suppose, for example, that algorithms SE,SD only depend on the first half of the bits of their k-bit key. This is consistent with their being IND-CCA secure (in the sense that, if there exists an IND-CCA-secure symmetric encryption scheme, there also exists one with this property), but now, any test has probability at most $2^{-k/2}$ of being able to differentiate between K and a key $L\neq K$ that agrees with K in its first half.

$L \neq K$ and return a random answer, while if dec $= 1$, it can output gs as its guess to challenge bit b and halt.

A precise description and analysis of $\mathsf{KeyTest}$ are in Appendix A, but we briefly sketch the ideas here. The algorithm has two phases. In the first phase, it repeatedly tests whether or not

$$\mathsf{SD}^{R_s}(L, \mathsf{SE}^{R_s}(K, \mathrm{LR}(T_0, T_0, b))) \;=\; T_0 \quad \text{and} \quad \mathsf{SD}^{R_s}(L, \mathsf{SE}^{R_s}(K, \mathrm{LR}(T_1, T_1, b))) \;=\; T_1 \;,$$

where T_0, T_1 are some distinct "test" messages. If any of these checks fails, it knows that $L \neq K$ and returns $(0, 0)$. (However, the checks can succeed with high probability even if $L \neq K$.) In the next phase, it repeatedly computes $\mathsf{SD}^{R_s}(L, \mathsf{SE}^{R_s}(K, \mathrm{LR}(T_0, T_1, b)))$ and, if *all* these computations yield T_{gs} for some bit gs, it returns $(1, \mathrm{gs})$. The analysis shows that, conditional on the first phase not returning $(0, 0)$, the bit gs from the second stage equals b with overwhelming probability.

A subtle point arises with relation to the test. Recall that $\boldsymbol{H}$ is making queries to $\mathsf{SD}^{R_s}(K, \cdot)$. $\boldsymbol{S}$ will answer these via its own oracle of the same name. Now, consider the event that $\boldsymbol{H}$ queries to $\mathsf{SD}^{R_s}(K, \cdot)$ a ciphertext C generated in some execution of $\mathsf{KeyTest}$. If $\boldsymbol{S}$ calls $\mathsf{SD}^{R_s}(K, C)$ to obtain the answer, it would immediately become an illegitimate adversary and thus forgo its advantage, since C is a result of a call to $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot, \cdot, b))$ made by $\boldsymbol{S}$ via subroutine $\mathsf{KeyTest}$. There are a few ways around this, and the one we use is to choose the initial "test" messages randomly so that $\boldsymbol{H}$ has low probability of being able to query a ciphertext C generated in some execution of $\mathsf{KeyTest}$.

We note that one might consider an alternative solution to $\boldsymbol{S}$'s problem of wanting to "detect" query K to H. Namely, reply to queries to H at random, then, after $\boldsymbol{H}$ terminates, pick one such query L at random, decrypt a challenge ciphertext via L, and use that to predict the challenge bit. Unfortunately, even though $L = K$ with probability $1/\operatorname{poly}(k)$, the advantage over one-half obtained by $\boldsymbol{S}$ via the strategy just outlined could be negligible because the wrong answers from the wrong random choices could overwhelm the right answer that arises when K is chosen.

We provide all the details and justify Equation (2) in Appendix A.

4 Uninstantiability of the Hash ElGamal scheme

In this section we show (cf. Theorem 4.1) that the RO-model Hash ElGamal scheme admits no IND-CCA-preserving instantiation. Below we begin by detailing what we mean by instantiation of a RO-model asymmetric encryption scheme. This will refer to a RO-model scheme which, as per the formal definitions in Section 2, uses a single random oracle mapping $\{0,1\}^*$ to $\{0,1\}$.

INSTANTIATING RO-MODEL ASYMMETRIC ENCRYPTION SCHEMES. A *poly-time family of functions* $\overline{F}$ associates to security parameter $k \in \mathbb{N}$ and key $fk \in \{0,1\}^{\mathrm{FKL}(k)}$ a map $\overline{F}^k(fk, \cdot)\colon \{0,1\}^* \to \{0,1\}$. The *key length* FKL of the family of functions is a polynomial in k. We require that there exist a polynomial t such that $\overline{F}^k(fk, x)$ is computable in $t(k + |x|)$ time for all $k \in \mathbb{N}$, $fk \in \{0,1\}^{\mathrm{FKL}(k)}$ and $x \in \{0,1\}^*$.

An *instantiation* of a RO-model asymmetric encryption scheme $\mathsf{AS} = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ via family $\overline{F}$ is the standard-model asymmetric encryption scheme $\overline{\mathsf{AS}} = (\overline{\mathsf{AK}}, \overline{\mathsf{AE}}, \overline{\mathsf{AD}})$ whose constituent algorithms are illustrated in Figure 2. As these indicate, the public and secret keys of the original scheme are enhanced to also include a key fk specifying the function $\overline{F}^k(fk, \cdot)$, and calls to the random oracle are then replaced by evaluations of this function in all algorithms.

THE UNINSTANTIABILITY RESULT. The formal statement of the result is the following.

$\overline{\mathsf{AK}}(1^k)$	$\overline{\mathsf{AE}}(\overline{pk}, K)$	$\overline{\mathsf{AD}}(\overline{sk}, C)$
$fk \xleftarrow{\$} \{0,1\}^{\text{FKL}(k)}$ $(pk, sk) \xleftarrow{\$} \mathsf{AK}^{\overline{F}^k(fk,\cdot)}(1^k)$ Return $((pk, fk), (sk, fk))$	Parse $\overline{pk}$ as (pk, fk) $C \xleftarrow{\$} \mathsf{AE}^{\overline{F}^k(fk,\cdot)}(pk, K)$ Return C	Parse $\overline{sk}$ as (sk, fk) $K \leftarrow \mathsf{AD}^{\overline{F}^k(fk,\cdot)}(sk, C)$ Return K

Figure 2: Algorithms of the standard-model asymmetric encryption scheme $\overline{\mathsf{AS}} = (\overline{\mathsf{AK}}, \overline{\mathsf{AE}}, \overline{\mathsf{AD}})$ obtained by instantiating RO-model asymmetric encryption scheme $\mathsf{AS} = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ via poly-time family of functions $\overline{F}$.

Theorem 4.1 Let $\mathsf{HEG}[\mathsf{CG}] = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ be the RO-model Hash ElGamal scheme associated to a cyclic-group generator CG. Let $\overline{\mathsf{HEG}}[\mathsf{CG}] = (\overline{\mathsf{AK}}, \overline{\mathsf{AE}}, \overline{\mathsf{AD}})$ be *any* instantiation of $\mathsf{HEG}[\mathsf{CG}]$ via a poly-time family of functions. Then $\overline{\mathsf{HEG}}[\mathsf{CG}]$ is *not* IND-CCA preserving. ▌

Proof of Theorem 4.1. Let $\overline{F}$ be the poly-time family of functions used in $\overline{\mathsf{HEG}}[\mathsf{CG}]$ to replace the random oracle. We will construct an IND-CCA-secure symmetric encryption scheme SS such that the mm-hybrid encryption scheme $(\overline{\mathsf{HEG}}[\mathsf{CG}], \mathsf{SS})$ is not IND-CCA secure. This proves the theorem.

Let us say that a value $\overline{pk}$ is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$*-valid public key* if there exists a value $\overline{sk}$ such that $(\overline{pk}, \overline{sk}) \in [\overline{\mathsf{AK}}(1^k)]$. We first define two polynomial-time algorithms VfPK and $\mathsf{VfCtxt}_{\overline{F}}$ which are used by SS.

Algorithm VfPK, which we call a *key verifier*, takes inputs 1^k and $\overline{pk}$, and outputs 1 if and only if $\overline{pk}$ is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key. The algorithm works by parsing $\overline{pk}$ as $((q, g, X), fk)$, where $fk \in \{0,1\}^{\text{FKL}}$, and then returning 1 if and only if q and $2q+1$ are primes, g is a generator of the order q cyclic subgroup $\langle g \rangle$ of $\mathbb{Z}^*_{2q+1}$, $|2q+1| = k$, and $X \in \langle g \rangle$. This algorithm can be implemented in polynomial-time based on standard facts from computational number theory, and even deterministically, given the existence of polynomial-time primality tests [1]. We omit the details.

Algorithm $\mathsf{VfCtxt}_{\overline{F}}$, which we call a *ciphertext verifier*, takes inputs $1^k, \overline{pk}, K, C$, where $\overline{pk}$ is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key and $K \in \{0,1\}^k$. It runs $\overline{\mathsf{AE}}(\overline{pk}, K)$ and outputs 1 if the result is C, and 0 otherwise. In other words, $\mathsf{VfCtxt}_{\overline{F}}$ verifies whether C is indeed an encryption of message K under the given public key $\overline{pk}$. This is possible because the encryption algorithm AE of $\mathsf{HEG}[\mathsf{CG}]$ (cf. Figure 1), and hence the encryption algorithm $\overline{\mathsf{AE}}$ of $\overline{\mathsf{HEG}}[\mathsf{CG}]$, is deterministic.

Let $\mathsf{SS}' = (\mathsf{SK}', \mathsf{SE}', \mathsf{SD}')$ be any standard-model IND-CCA-secure symmetric encryption scheme. (Recall an implicit assumption is that some such scheme exists, since otherwise *all* asymmetric encryptions schemes are by default IND-CCA preserving and the entire problem we are considering is moot.) The construction of SS is in terms of SS' and algorithms VfPK and $\mathsf{VfCtxt}_{\overline{F}}$. We use the notation $\langle(\cdot,\cdot)\rangle$ to denote an injective, polynomial-time computable encoding of pairs of strings as strings such that given $\langle(M_1, M_2)\rangle$, M_1 and M_2 can be recovered in polynomial time. If s is a string and $a \le b$ are integers then $s[a \ldots b]$ denotes the string consisting of bit positions a through b of s. The algorithms constituting $\mathsf{SS} = (\mathsf{SK}, \mathsf{SE}, \mathsf{SD})$ are depicted in Figure 3. To conclude the proof, we need only establish the following propositions.

Proposition 4.2 Symmetric encryption scheme SS is IND-CCA secure.

Proposition 4.3 Multi-message hybrid encryption scheme $(\overline{\mathsf{HEG}}[\mathsf{CG}], \mathsf{SS})$ is not IND-CCA secure.

$\mathsf{SK}(1^k)$
$K' \xleftarrow{\$} \mathsf{SK}'(1^{\lceil k/2 \rceil})$
$K_2 \xleftarrow{\$} \{0,1\}^{\lfloor k/2 \rfloor}$
Return $K' \| K_2$
	$\mathsf{SE}(K, M)$
	$k \leftarrow \|K\|$
	$K' \leftarrow K[1 \ldots \lceil k/2 \rceil]$
	$K_2 \leftarrow K[1 + \lceil k/2 \rceil \ldots k]$
	$C' \leftarrow \mathsf{SE}'(K', M)$
	Parse M as $\langle (M_1, M_2) \rangle$
	If the parsing fails then
	Return $C' \| 1$ EndIf
	$p \leftarrow \mathsf{VfPK}(1^k, M_1)$
	$c \leftarrow \mathsf{VfCtxt}_{\overline{F}}(1^k, M_1, K, M_2)$
	If $(p = 1$ and $c = 1)$ then
	Return $C' \| 0$
	else Return $C' \| 1$ EndIf
		$\mathsf{SD}(K, C)$
		$k \leftarrow \|K\|$
		$K' \leftarrow K[1 \ldots \lceil k/2 \rceil]$
		$K_2 \leftarrow K[1 + \lceil k/2 \rceil \ldots k]$
		Parse C as $C' \| d$, where $d \in \{0,1\}$
		$M' \leftarrow \mathsf{SD}'(K', C')$
		Parse M' as $\langle (M_1, M_2) \rangle$
		If the parsing fails then
		If $d = 1$ then Return M'
		else Return $\perp$ EndIf
		$p \leftarrow \mathsf{VfPK}(1^k, M_1)$
		$c \leftarrow \mathsf{VfCtxt}_{\overline{F}}(1^k, M_1, K, M_2)$
		If $(d = 0$ and $p = 1$ and $c = 1)$ then
		Return M' EndIf
		If $(d = 1$ and $(p \neq 1$ or $c \neq 1))$ then
		Return M' EndIf
		Return $\perp$

Figure 3: Algorithms of the symmetric encryption scheme $\mathsf{SS} = (\mathsf{SK}, \mathsf{SE}, \mathsf{SD})$ for the proof of Theorem 4.1. Above, $\langle (M_1, M_2) \rangle$ denotes an encoding of the pair of strings (M_1, M_2) as a string.

Proof of Proposition 4.2: Let us first provide some intuition. Note that on input M, encryption algorithm $\mathsf{SE}(K_1' \| K_2, \cdot)$ uses the encryption algorithm SE' of an IND-CCA-secure scheme to compute $C' \xleftarrow{\$} \mathsf{SE}'(K_1', M)$ and outputs $C' \| 0$ or $C' \| 1$, depending on whether M has some "special" form or not. The ciphertext ends with 0 if M parses as a pair (M_1, M_2) such that algorithms $\mathsf{VfPK}, \mathsf{VfCtxt}_{\overline{F}}$ indicate that M_1 is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key and $M_2 \in [\overline{\mathsf{AE}}(M_1, K_1' \| K_2)]$. The decryption algorithm $\mathsf{SD}(K_1' \| K_2, \cdot)$ on input $C' \| d$, where d is a bit, computes $M' \leftarrow \mathsf{SD}'(K_1', C')$ and returns M' only if either M' is of the special form and $d = 0$, or M' is not of this form and $d = 1$. Therefore, an obvious strategy for an adversary against SS is to query its oracle $\mathsf{SE}(K, \mathrm{LR}(\cdot, \cdot, b))$ on a pair of messages such that one of them is of this special form and the other is not. Using the unique decryptability of $\overline{\mathsf{AE}}$ and the fact that K_2 is chosen at random, independently from the adversary's view, we show that it cannot find such queries except with negligible probability. Moreover, we show that any strategy for the adversary can be employed by an attacker against scheme SS' to win its game. Details follow.

Let $\boldsymbol{S}$ be a legitimate polynomial-time adversary attacking SS. We will construct a legitimate polynomial-time adversary $\boldsymbol{S}'$ such that

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS}, \boldsymbol{S}}(k) \;\leq\; \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS}', \boldsymbol{S}'}(\lceil k/2 \rceil) + \frac{O(Q(k))}{2^{\lfloor k/2 \rfloor}}\,, \tag{5}$$

where Q is a polynomial upper bounding the total number of queries made by $\boldsymbol{S}$ to its different oracles. Since SS' is assumed IND-CCA secure, the advantage function associated to $\boldsymbol{S}'$ above is negligible, and thus so is the advantage function associated to $\boldsymbol{S}$. To complete the proof, we need to specify adversary $\boldsymbol{S}'$ and prove Equation (5).

Adversary $\boldsymbol{S}'$ is given input $1^{\lceil k/2 \rceil}$ and has access to oracles $\mathsf{SE}'(K_1', \mathrm{LR}(\cdot, \cdot, b))$ and $\mathsf{SD}'(K_1', \cdot)$. Its goal is to guess the bit b. It runs $\boldsymbol{S}$ on input 1^k. In this process, $\boldsymbol{S}$ will query its two oracles $\mathsf{SE}(K, \mathrm{LR}(\cdot, \cdot, b))$ and $\mathsf{SD}(K, \cdot)$. To answer a query to the first of these oracles, $\boldsymbol{S}'$ forwards the query to its oracle $\mathsf{SE}'(K_1', \mathrm{LR}(\cdot, \cdot, b))$, appends 1 to the oracle's reply and returns the result to $\boldsymbol{S}$.

To answer a query to the second oracle, $\boldsymbol{S}'$ checks the last bit of the query. If it is 0, $\boldsymbol{S}'$ returns $\perp$ to $\boldsymbol{S}$. Otherwise, it removes the last bit, forwards the result to its oracle $\mathsf{SD}'(K'_1, \cdot)$, and returns the answer to $\boldsymbol{S}$. When $\boldsymbol{S}$ outputs its guess b', $\boldsymbol{S}'$ returns b'.

We now analyze $\boldsymbol{S}'$. Consider the experiment in which $\boldsymbol{S}'$ attacks SS'. We define the following events.

$\mathsf{Succ}(\boldsymbol{S}')$	:	$\boldsymbol{S}'$ is successful, meaning its output equals the challenge bit b
BadE	:	$\boldsymbol{S}$ makes a query to oracle $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$ in which one of the messages can be parsed as $\langle (M_1, M_2) \rangle$ such that M_1 is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key and $M_2 \in [\overline{\mathsf{AE}}(M_1, K)]$
BadD	:	$\boldsymbol{S}$ makes a query to oracle $\mathsf{SD}(K, \cdot)$ that can be parsed as $C' \| d$, where d is a bit, such that $\mathsf{SD}'(K'_1, C') = \langle (M_1, M_2) \rangle$, where M_1 is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key and $M_2 \in [\overline{\mathsf{AE}}(M_1, K)]$

For the experiment in which $\boldsymbol{S}$ attacks SS, we define the following event.

$\mathsf{Succ}(\boldsymbol{S})$: $\boldsymbol{S}$ is successful, meaning its output equals the challenge bit b

We claim that if events BadE and BadD do not occur, then $\boldsymbol{S}'$ simulates perfectly the environment provided to $\boldsymbol{S}$ in its attack against SS. First, note that answers to queries to oracle $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$ can only be off by the last bit. In the absence of the "bad" events, each ciphertext returned to $\boldsymbol{S}$ as a reply to a query to oracle $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$ has 1 as the last bit. This is also the case in $\boldsymbol{S}$'s real attack. If $\boldsymbol{S}$ queries $\mathsf{SD}(K, \cdot)$ with a ciphertext $C'\|0$, assuming events BadE and BadD do not occur, $\boldsymbol{S}'$ gives $\boldsymbol{S}$ the response it would get in the real attack, namely $\perp$. Since $\boldsymbol{S}$ is legitimate, if it queries oracle $\mathsf{SD}(K, \cdot)$ with a ciphertext $C'\|1$, then C' must not have previously been returned by oracle $\mathsf{SE}'(K'_1, \mathrm{LR}(\cdot,\cdot,b))$. Thus $\boldsymbol{S}'$ can legitimately make query C' to its oracle $\mathsf{SD}'(K'_1, \cdot)$. If M is the response, then, assuming that events BadE and BadD do not occur, the answer $\boldsymbol{S}$ expects is exactly M. Therefore,

$$
\begin{aligned}
\Pr\left[\,\mathsf{Succ}(\boldsymbol{S}')\,\right] &\geq \Pr\left[\,\mathsf{Succ}(\boldsymbol{S}') \mid \neg\mathsf{BadE} \wedge \neg\mathsf{BadD}\,\right] - \Pr\left[\,\mathsf{BadE} \vee \mathsf{BadD}\,\right] \\
&\geq \Pr\left[\,\mathsf{Succ}(\boldsymbol{S})\,\right] - \Pr\left[\,\mathsf{BadE} \vee \mathsf{BadD}\,\right] .
\end{aligned}
$$

We now provide an upper bound for the probability of event $\mathsf{BadE} \vee \mathsf{BadD}$. Let $q_e(k)$ and $q_d(k)$ be the number of queries $\boldsymbol{S}$ makes to oracles $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$ and $\mathsf{SD}(K, \cdot)$, respectively, on input 1^k. We observe that if M_1 is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key, then for any $M_2 \in \{0,1\}^*$, there exists a unique $K' \in [\mathsf{SK}(1^k)]$ such that $M_2 \in [\overline{\mathsf{AE}}(M_1, K')]$. Recall that the key for oracles $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$ and $\mathsf{SD}(K, \cdot)$ is $K = K'_1 \| K_2$, where K_2 is chosen uniformly at random from $\{0,1\}^{\lfloor k/2 \rfloor}$ and is independent from $\boldsymbol{S}$'s view. Therefore, for any query made by $\boldsymbol{S}$ to oracle $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$, the probability that one of the messages in the query parses as $\langle (M_1, M_2) \rangle$ such that M_1 is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key and $M_2 \in [\overline{\mathsf{AE}}(M_1, K)]$ is at most $2/2^{\lfloor k/2 \rfloor}$. Similarly, for any query $C'\|d$, where d is a bit, made by $\boldsymbol{S}$ to oracle $\mathsf{SD}(K, \cdot)$, the probability that $\mathsf{SD}'(K'_1, C') = M'$, where M' parses as $\langle (M_1, M_2) \rangle$, M_1 is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key and $M_2 \in [\overline{\mathsf{AE}}(M_1, K)]$ is at most $1/2^{\lfloor k/2 \rfloor}$. Therefore,

$$
\Pr\left[\,\mathsf{BadE} \vee \mathsf{BadD}\,\right] \leq \frac{2q_e(k) + q_d(k)}{2^{\lfloor k/2 \rfloor}} \leq \frac{2 \cdot Q(k)}{2^{\lfloor k/2 \rfloor}} ,
$$

where $Q(k) = q_e(k) + q_d(k)$. Hence

$$
\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS}', \boldsymbol{S}'}(\lceil k/2 \rceil) = 2 \cdot \Pr\left[\,\mathsf{Succ}(\boldsymbol{S}')\,\right] - 1 \geq 2 \cdot \left(\Pr\left[\,\mathsf{Succ}(\boldsymbol{S})\,\right] - \frac{O(Q(k))}{2^{\lfloor k/2 \rfloor}} \right) - 1
$$

$$= \ \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) - \frac{O(Q(k))}{2^{\lfloor k/2 \rfloor}} \ .$$

Rearranging terms gives Equation (5). ▮

Proof of Proposition 4.3: We define a hybrid adversary $\boldsymbol{H}$ attacking $(\overline{\mathsf{HEG}}[\mathsf{CG}], \mathsf{SS})$. $\boldsymbol{H}$ is given inputs $\overline{pk} = ((q,g,X), fk)$ and C_a and has access to oracles $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{SD}(K,\cdot)$, and $\overline{\mathsf{AD}}(\overline{sk},\cdot)$, where $\overline{sk} = ((q,g,x), fk)$. Its goal is to guess the challenge bit b. By the definition of experiment $\mathbf{Exp}^{\text{ind-cca}}_{\overline{\mathsf{HEG}}[\mathsf{CG}],\mathsf{SS},\boldsymbol{H}}(k)$, $\overline{pk}$ is a $(\overline{\mathsf{HEG}}[\mathsf{CG}], k)$-valid public key and $C_a \in [\overline{\mathsf{AE}}(\overline{pk}, K)]$. Therefore, $\langle(\overline{pk}, C_a)\rangle$ is a message which, when encrypted with $\mathsf{SE}(K,\cdot)$, yields a ciphertext that has last bit 0. We observe that for any string C chosen at random from $\{0,1\}^{|C_a|} \setminus \{C_a\}$, the probability that $K = \overline{\mathsf{AD}}(\overline{sk}, C)$ is 0 (since $\overline{\mathsf{AE}}(\overline{pk}, K) = C_a$ and $\overline{\mathsf{AE}}$ is deterministic), i.e., the probability that $C \in [\overline{\mathsf{AE}}(\overline{pk}, K)]$ is 0. Hence $\langle(\overline{pk}, C)\rangle$ is a message which, when encrypted with $\mathsf{SE}(K,\cdot)$, yields a ciphertext that has last bit 1. (If $C \notin [\overline{\mathsf{AE}}(\overline{pk}, K)]$, then the last bit will be 1.) Thus, adversary $\boldsymbol{H}$ can construct two messages for which it can guess with probability 1 the last bit of the corresponding ciphertext. Using this information it can then guess the challenge bit. Details follow.

Adversary $\boldsymbol{H}$ chooses C at random from $\{0,1\}^{|C_a|} \setminus \{C_a\}$, makes a query $\langle(\overline{pk}, C_a)\rangle, \langle(\overline{pk}, C)\rangle$ to oracle $\mathsf{SE}(K, \mathrm{LR}(\cdot,\cdot,b))$, parses the response as $C'||d$, where d is a bit, and returns d. The running time of $\boldsymbol{H}$ is clearly polynomial in k. We claim that $\mathsf{Adv}^{\text{ind-cca}}_{\overline{\mathsf{HEG}}[\mathsf{CG}],\mathsf{SS},\boldsymbol{H}}(k) = 1$. To prove this, we consider the event

$\mathsf{Succ}(\boldsymbol{H})$: $\boldsymbol{H}$ is successful, meaning its output equals the challenge bit b

If challenge bit b is 0, then the response to $\boldsymbol{H}$'s query is a ciphertext that has last bit 0. If bit b is 1, then the response is a ciphertext that has last bit 1. Thus

$$\Pr[\,\mathsf{Succ}(\boldsymbol{H})\,] \ = \ \frac{1}{2} + \frac{1}{2} \ = \ 1 \ .$$

Hence

$$\mathsf{Adv}^{\text{ind-cca}}_{\overline{\mathsf{HEG}}[\mathsf{CG}],\mathsf{SS},\boldsymbol{H}}(k) \ = \ 2 \cdot \Pr[\,\mathsf{Succ}(\boldsymbol{H})\,] - 1 \ = \ 1 \,,$$

as desired. ▮

Notice that the adversary constructed in the proof of Proposition 4.3 does not make any queries to its oracles $\mathsf{SD}(K,\cdot)$ and $\overline{\mathsf{AD}}(\overline{sk},\cdot)$.

Remark 4.4 An interesting question at this point may be why the proof of Theorem 4.1 fails for the RO-model Hash ElGamal scheme $\mathsf{HEG}[\mathsf{CG}]$ associated to a cyclic-group generator CG —it must, since otherwise Theorem 3.1 would be contradicted— but succeeds for any instantiation of this scheme. The answer is that symmetric encryption scheme SS, depicted in Figure 3 runs a ciphertext verifier $\mathsf{VfCtxt}_{\overline{F}}$ for the asymmetric encryption scheme in question. In the case of the RO-model scheme $\mathsf{HEG}[\mathsf{CG}]$, any ciphertext verifier must query random oracles G and H. But as we clarified in Section 2, SS does not have access to these oracles (although it might have access to its own, independently chosen oracle R_s), and so cannot run such a ciphertext verifier. The adversary of course does have access to G, H, but has no way to "pass" these objects to the encryption algorithm of the symmetric encryption scheme. On the other hand, in the instantiated scheme, the keys describing the functions instantiating the random oracles may be passed by the adversary to the encryption algorithm of SS in the form of a message containing the public key, giving SS the ability to run the ciphertext verifier. This might lead one to ask why SS does not have oracle access to G, H. This is explained in Remark 2.1. ∎

5 A generalization

In this section, we identify a subclass of IND-CCA-preserving asymmetric encryption schemes that we call key-verifiable, ciphertext-verifiable IND-CCA-preserving asymmetric encryption schemes. We show that such schemes exist in the RO model, but do not exist in the standard model. We then discuss how this generalizes our results about the Hash El Gamal scheme. We begin by defining the two properties mentioned above, namely, key verifiability and ciphertext verifiability.

Let $\mathsf{AS} = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ be an asymmetric encryption scheme. Let us say that a value pk is an (AS, k)-*valid public key* if there exists a value sk such that $(pk, sk) \in [\mathsf{AK}(1^k)]$. We say that AS is *key verifiable* if there exists a polynomial-time, possibly randomized algorithm VfPK (called the *key verifier*) and a negligible function ν (called the *error probability* of VfPK) such that $\mathsf{VfPK}(1^k, pk)$ returns 1 with probability at least $1 - \nu(k)$ if pk is an (AS, k)-valid public key, and returns 1 with probability at most $\nu(k)$ otherwise. If AK has access to a random oracle, then VfPK is given access to the same random oracle.

We say that asymmetric encryption scheme $\mathsf{AS} = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ is *ciphertext verifiable* if there exists a polynomial-time, possibly randomized algorithm VfCtxt (called the *ciphertext verifier*) and a negligible function ν (called the *error probability* of VfCtxt) such that, if VfCtxt is run on inputs $1^k, pk, K, C$, where pk is an (AS, k)-valid public key and $K \in \{0,1\}^k$, then VfCtxt returns 1 with probability at least $1 - \nu(k)$ if $C \in [\mathsf{AE}(pk, K)]$, and returns 1 with probability at most $\nu(k)$ otherwise. If AE or AD access a random oracle, then VfCtxt is given access to the same random oracle.

The following result will be used later.

Proposition 5.1 Suppose AS is a RO-model asymmetric encryption scheme that is both key verifiable and ciphertext verifiable. Let $\overline{\mathsf{AS}}$ be any instantiation of AS via a poly-time family of functions. Then $\overline{\mathsf{AS}}$ is also both key verifiable and ciphertext verifiable. ∎

Proof of Proposition 5.1: Let VfPK and VfCtxt be a key verifier and a ciphertext verifier for AS, respectively. Let $\overline{F}$ be the poly-time family of functions used in $\overline{\mathsf{AS}}$ to replace the random oracle. Recall that a public key of $\overline{\mathsf{AS}}$ contains a public key pk of AS and also a key fk specifying an instance of $\overline{F}$. We define algorithms $\overline{\mathsf{VfPK}}_{\overline{F}}$ and $\overline{\mathsf{VfCtxt}}_{\overline{F}}$.

On inputs $1^k, s$, $\overline{\mathsf{VfPK}}_{\overline{F}}$ attempts to parse s as a pair (pk, fk). If it fails, it returns 0. Otherwise, it runs $\mathsf{VfPK}(1^k, pk)$. If the result is 0, it returns 0. Otherwise, it verifies that $fk \in \{0,1\}^{\mathrm{FKL}(k)}$. If so, it returns 1, if not it returns 0. Clearly, $\overline{\mathsf{VfPK}}_{\overline{F}}$ is a key verifier for $\overline{\mathsf{AS}}$.

$\overline{\mathsf{VfCtxt}}_{\overline{F}}$ is identical to VfCtxt except that the random oracle is replaced with the same instance of $\overline{F}$ used in $\overline{\mathsf{AS}}$ to replace the oracle. ∎

We now observe that, in the RO model, there exist key-verifiable, ciphertext-verifiable IND-CCA-preserving asymmetric encryption schemes, meaning the goal of key-verifiable, ciphertext-verifiable asymmetric encryption is achievable in this model.

Theorem 5.2 Suppose there exists a cyclic-group generator for which the CDH problem is hard. Then there exists a key-verifiable, ciphertext-verifiable RO-model asymmetric encryption scheme that is IND-CCA-preserving in the RO model. ∎

Proof of Theorem 5.2: If the CDH problem is hard for cyclic-group generator CG, then Theorem 3.1 guarantees that the associated Hash ElGamal asymmetric encryption scheme $\mathsf{HEG}[\mathsf{CG}]$, defined in Section 3, is IND-CCA preserving in the RO model. The proof of Theorem 3.1 defines a key verifier VfPK and a ciphertext verifier VfCtxt for $\mathsf{HEG}[\mathsf{CG}]$, each having error probability 0. ▮

Next, we show that in the standard model, there do *not* exist key-verifiable, ciphertext-verifiable IND-CCA-preserving asymmetric encryption schemes, meaning the goal of key-verifiable, ciphertext-verifiable asymmetric encryption is *not* achievable in this model.

Theorem 5.3 Let $\overline{\mathsf{AS}}$ be a standard-model asymmetric encryption scheme that is both key verifiable and ciphertext verifiable. Then $\overline{\mathsf{AS}}$ is *not* IND-CCA preserving. ▮

Theorem 5.3 is proved below. We first state and prove our final result.

Theorem 5.4 Let AS be a RO-model asymmetric encryption scheme that is both key verifiable and ciphertext verifiable. Let $\overline{\mathsf{AS}}$ be *any* instantiation of AS via a poly-time family of functions. Then $\overline{\mathsf{AS}}$ is *not* IND-CCA preserving. ▮

Proof of Theorem 5.4: $\overline{\mathsf{AS}}$ is a standard-model asymmetric encryption scheme. Proposition 5.1 implies that it inherits the key verifiability and ciphertext verifiability of AS. Theorem 5.3 then implies that it is not IND-CCA preserving. ▮

Note that Theorem 5.4 implies Theorem 4.1 because the Hash El Gamal scheme is a RO-model scheme that is key verifiable and ciphertext verifiable. Theorem 5.4 is, however, more general, and shows that the uninstantiability of the Hash El Gamal scheme arises not due to some "peculiar" use of random oracles, but due to the fact that the scheme possesses the properties of key verifiability and ciphertext verifiability.

Proof of Theorem 5.3. The proof is almost identical to the proof of Theorem 4.1. Accordingly, we use the same notation and the previous results, and only indicate the differences. Let VfPK and VfCtxt be a key verifier and a ciphertext verifier for $\overline{\mathsf{AS}}$, respectively. The main difference is that now VfPK and VfCtxt can be randomized algorithms with non-zero error probabilities.

We present an IND-CCA-secure symmetric encryption scheme SS such that the mm-hybrid encryption scheme $(\overline{\mathsf{AS}}, \mathsf{SS})$ is not IND-CCA secure. This proves the theorem.

Let $\mathsf{SS}' = (\mathsf{SK}', \mathsf{SE}', \mathsf{SD}')$ be any standard-model IND-CCA-secure symmetric encryption scheme. The construction of SS is in terms of SS' and algorithms VfPK and VfCtxt, and is exactly as in the proof of Theorem 4.1. See Figure 3. To conclude the proof, we need only establish the following propositions.

Proposition 5.5 Symmetric encryption scheme SS is IND-CCA secure.

Proposition 5.6 Multi-message hybrid encryption scheme $(\overline{\mathsf{AS}}, \mathsf{SS})$ is not IND-CCA secure.

Proof of Proposition 5.5: Let $\boldsymbol{S}$ be a legitimate polynomial-time adversary attacking SS. We will construct a legitimate polynomial-time adversary $\boldsymbol{S}'$ such that

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) \;\leq\; \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS}',\boldsymbol{S}'}(\lceil k/2 \rceil) + \frac{O(Q(k))}{2^{\lfloor k/2 \rfloor}} + O(Q(k)) \cdot \nu(k)\,, \tag{6}$$

where Q is a polynomial upper bounding the total number of queries made by $\boldsymbol{S}$ to its different oracles, and ν is a negligible function related to the error probabilities of algorithms VfPK and

VfCtxt. Note that the last term is the only difference with Equation (5). Since SS' is assumed IND-CCA secure, the advantage function associated to $\boldsymbol{S}'$ above is negligible, and thus so is the advantage function associated to $\boldsymbol{S}$. To complete the proof, we need to specify adversary $\boldsymbol{S}'$ and prove Equation (6).

Adversary $\boldsymbol{S}'$ is identical to the adversary in the proof of Proposition 4.2. The analysis of $\boldsymbol{S}'$ is similar, but we need to take into account the possibility that algorithms VfPK and VfCtxt err. For this reason, for the experiment in which $\boldsymbol{S}$ attacks SS, we define the following additional event.

Crct : Every time algorithms VfPK and VfCtxt are invoked, they return the correct value

We claim that if events BadE and BadD do not occur, then $\boldsymbol{S}'$ simulates perfectly the environment provided to $\boldsymbol{S}$ in its attack against SS when algorithms VfPK and VfCtxt never err. First, note that answers to queries to oracle $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ can only be off by the last bit. In the absence of the "bad" events, each ciphertext returned to $\boldsymbol{S}$ as a reply to a query to oracle $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ has 1 as the last bit. This is also the case in $\boldsymbol{S}$'s real attack when algorithms VfPK and VfCtxt are always correct. If $\boldsymbol{S}$ queries $\mathsf{SD}(K,\cdot)$ with a ciphertext $C'||0$, assuming events BadE and BadD do not occur, $\boldsymbol{S}'$ gives $\boldsymbol{S}$ the response it would get in the real attack when algorithms VfPK and VfCtxt are always correct, namely $\perp$. Since $\boldsymbol{S}$ is legitimate, if it queries oracle $\mathsf{SD}(K,\cdot)$ with a ciphertext $C'||1$, then C' must not have previously been returned by oracle $\mathsf{SE}'(K_1', \mathrm{LR}(\cdot,\cdot,b))$. Thus $\boldsymbol{S}'$ can legitimately make query C' to its oracle $\mathsf{SD}'(K_1',\cdot)$. If M is the response, then, assuming that events BadE and BadD do not occur, the answer $\boldsymbol{S}$ expects when algorithms VfPK and VfCtxt are always correct is exactly M. Therefore,

$$\begin{aligned}
\Pr\left[\,\mathsf{Succ}(\boldsymbol{S}')\,\right] &\geq \Pr\left[\,\mathsf{Succ}(\boldsymbol{S}') \mid \neg\mathsf{BadE} \wedge \neg\mathsf{BadD}\,\right] - \Pr\left[\,\mathsf{BadE} \vee \mathsf{BadD}\,\right] \\
&\geq \Pr\left[\,\mathsf{Succ}(\boldsymbol{S}) \mid \mathsf{Crct}\,\right] - \Pr\left[\,\mathsf{BadE} \vee \mathsf{BadD}\,\right] \\
&\geq \Pr\left[\,\mathsf{Succ}(\boldsymbol{S})\,\right] - \Pr\left[\,\neg\mathsf{Crct}\,\right] - \Pr\left[\,\mathsf{BadE} \vee \mathsf{BadD}\,\right].
\end{aligned}$$

We now provide an upper bound for the probability of event $\neg$Crct. (The bound for $\mathsf{BadE} \vee \mathsf{BadD}$ is identical to the one in the proof of Proposition 4.2.) Let $q_e(k)$ and $q_d(k)$ be the number of queries $\boldsymbol{S}$ makes to oracles $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ and $\mathsf{SD}(K,\cdot)$, respectively, on input 1^k. Let ν_1 be the error probability of key verifier VfPK, and ν_2 the error probability of ciphertext verifier VfCtxt. Then

$$\Pr\left[\,\neg\mathsf{Crct}\,\right] \leq q_e(k)\cdot(\nu_1(k)+\nu_2(k)) + q_d(k)\cdot(\nu_1(k)+\nu_2(k)) = Q(k)\cdot\nu(k)\,,$$

where $Q(k) = q_e(k) + q_d(k)$ and $\nu(k) = \nu_1(k) + \nu_2(k)$.

Hence

$$\begin{aligned}
\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS}',\boldsymbol{S}'}(\lceil k/2 \rceil) &= 2\cdot\Pr\left[\,\mathsf{Succ}(\boldsymbol{S}')\,\right] - 1 \;\geq\; 2\cdot\left(\Pr\left[\,\mathsf{Succ}(\boldsymbol{S})\,\right] - Q(k)\cdot\nu(k) - \frac{O(Q(k))}{2^{\lfloor k/2 \rfloor}}\right) - 1 \\
&= \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) - O(Q(k))\cdot\nu(k) - \frac{O(Q(k))}{2^{\lfloor k/2 \rfloor}}\,.
\end{aligned}$$

Rearranging terms gives Equation (6). ▮

Proof of Proposition 5.6: We define a hybrid adversary $\boldsymbol{H}$ attacking $(\overline{\mathsf{AS}}, \mathsf{SS})$ exactly as in the proof of Proposition 4.3. We claim that $\mathsf{Adv}^{\text{ind-cca}}_{\overline{\mathsf{AS}},\mathsf{SS},\boldsymbol{H}}(k) \geq 1 - 2^{-k} - \nu(k)$, where ν is a negligible function related to the error probabilities of algorithms VfPK and VfCtxt. The analysis is similar to the one in the proof of Proposition 4.3, but we need to take into account the additional event

Crct : Every time algorithms VfPK and VfCtxt are invoked, they return the correct value, and the possibility that $K = \mathsf{AD}(sk, C)$, where C is chosen at random from $\{0,1\}^{|C_a|} \setminus \{C_a\}$. The latter can happen with probability at most 2^{-k}. I.e., the probability that $C \in [\overline{\mathsf{AE}}(pk, K)]$ is at most 2^{-k}. Hence $\langle (pk, C) \rangle$ is a message which, when encrypted with $\mathsf{SE}(K, \cdot)$, yields a ciphertext that with overwhelming probability has last bit 1. (If $C \notin [\overline{\mathsf{AE}}(pk, K)]$, then the last bit will be 1.) Assume that event Crct occurs. If challenge bit b is 0, then the response to $\boldsymbol{H}$'s query is a ciphertext that has last bit 0. If bit b is 1, then with probability at least $1 - 2^{-k}$, the response is a ciphertext that has last bit 1. Thus

$$\Pr[\,\mathsf{Succ}(\boldsymbol{H})\,] \;\geq\; \Pr[\,\mathsf{Succ}(\boldsymbol{H}) \mid \mathsf{Crct}\,] - \Pr[\,\neg\mathsf{Crct}\,] \;\geq\; \frac{1}{2}\cdot\left(1 - \frac{1}{2^k}\right) + \frac{1}{2} - \Pr[\,\neg\mathsf{Crct}\,]$$

If ν_1 is the error probability of key verifier VfPK, and ν_2 is the error probability of ciphertext verifier VfCtxt, then $\Pr[\,\neg\mathsf{Crct}\,] \;\leq\; \nu_1(k) + \nu_2(k)$. Hence

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) \;=\; 2\cdot\Pr[\,\mathsf{Succ}(\boldsymbol{H})\,] - 1 \;\geq\; 1 - 2^{-k} - 2\cdot(\nu_1(k) + \nu_2(k)) \;=\; 1 - 2^{-k} - \nu(k)\,,$$

where $\nu(k) = 2\cdot(\nu_1(k) + \nu_2(k))$. ▮

References

[1] M. Agarwal, N. Saxena and N. Kayal, "PRIMES is in P," *Preprint. Available at* `http://www.cse.iitk.ac.in/news/primality.html`, August 6, 2002.

[2] J. Baek, B. Lee and K. Kim, "Secure length-saving ElGamal encryption under the Computational Diffie-Hellman assumption," *Proceedings of the* 1900 *Australasian Conference on Information Security and Privacy– ACISP*, Lecture Notes in Computer Science Vol. 1841, E. Dawson, A. Clark and C. Boyd ed., Springer-Verlag, 1900.

[3] J. Baek, B. Lee and K. Kim, "Provably secure length-saving public-key encryption scheme under the computational Diffie-Hellman assumption," *ETRI Journal*, 22(4), 2000.

[4] M. Bellare, A. Desai, E. Jokipii and P. Rogaway, "A concrete security treatment of symmetric encryption: Analysis of the DES modes of operation," *Proceedings of the 38th Symposium on Foundations of Computer Science*, IEEE, 1997.

[5] M. Bellare and P. Rogaway, Random oracles are practical: a paradigm for designing efficient protocols, *First ACM Conference on Computer and Communications Security*, ACM, 1993.

[6] R. Canetti, U. Feige, O. Goldreich and M. Naor, "Adaptively secure multi-party computation," *Proceedings of the 28th Annual Symposium on the Theory of Computing*, ACM, 1996.

[7] R. Canetti, O. Goldreich, S. Halevi, "The random oracle methodology, revisited," *Proceedings of the 30th Annual Symposium on the Theory of Computing*, ACM, 1998.

[8] J.-S. Coron, H. Handschuh, M. Joye, P. Paillier, D. Pointcheval, C. Tymen, "GEM: A Generic Chosen-Ciphertext Secure Encryption Method", *Topics in Cryptology – CT-RSA '02*, Lecture Notes in Computer Science Vol. 2271, B. Preneel ed., Springer-Verlag, 2002.

[9] J.-S. Coron, H. Handschuh, M. Joye, P. Paillier, D. Pointcheval, C. Tymen, "Optimal Chosen-Ciphertext Secure Encryption of Arbitrary-Length Messages," *Proceedings of the Fifth International workshop on practice and theory in Public Key Cryptography (PKC'02)*, Lecture Notes in Computer Science Vol. 1431, D. Naccache and P. Paillier eds., Springer-Verlag, 2002.

[10] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," *Advances in Cryptology – CRYPTO '98*, Lecture Notes in Computer Science Vol. 1462, H. Krawczyk ed., Springer-Verlag, 1998.

[11] R. Cramer and V. Shoup, "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack," IACR ePrint archive Record 2001/108, 2001, `http://eprint.iacr.org/`.

[12] T. ElGamal, "A public key cryptosystem and signature scheme based on discrete logarithms," *IEEE Transactions on Information Theory*, vol 31, 1985.

[13] A. Fiat and A. Shamir, "How to prove yourself: practical solutions to identification and signature problems," *Advances in Cryptology – CRYPTO '86*, Lecture Notes in Computer Science Vol. 263, A. Odlyzko ed., Springer-Verlag, 1986.

[14] E. Fujisaki, T. Okamoto, "Secure Integration of Asymmetric and Symmetric Encryption Schemes," *Advances in Cryptology – CRYPTO '99*, Lecture Notes in Computer Science Vol. 1666, M. Wiener ed., Springer-Verlag, 1999.

[15] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Science*, Vol. 28, 1984, pp. 270–299.

[16] S. Goldwasser and Y. Taumann, "On the (in)security of the Fiat-Shamir paradigm," *Proceedings of the 44th Symposium on Foundations of Computer Science*, IEEE, 2003.

[17] S. Micali, "Computationally sound proofs," *SIAM Journal on Computing*, Vol. 30, No. 4, 2000, pp. 1253-1298.

[18] J. B. Nielsen "Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case," *Advances in Cryptology – CRYPTO '02*, Lecture Notes in Computer Science Vol. 2442, M. Yung ed., Springer-Verlag, 2002.

[19] T. Okamoto and D. Pointcheval "REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform," *Topics in Cryptology – CT-RSA '01*, Lecture Notes in Computer Science Vol. 2020, D. Naccache ed., Springer-Verlag, 2001.

[20] V. Shoup, "A proposal for an ISO standard for public key encryption", IACR ePrint archive Record 2001/112, 2001, `http://eprint.iacr.org/`.

A Proof of Theorem 3.1

We explained the ideas behind this proof in Section 3. Here we provide the full adversary constructions and analyses.

Proof setup. Let $\boldsymbol{H}$ be a polynomial-time hybrid adversary attacking $(\mathsf{AS}, \mathsf{SS})$. We will construct polynomial-time adversaries $\boldsymbol{S}$ and $\boldsymbol{C}$ such that

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) \;\leq\; \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) + O(Q(k)) \cdot \mathsf{Adv}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) + \frac{O(Q(k)^2)}{2^k}\,, \tag{7}$$

where $Q(k)$ is a polynomial upper bounding the number of queries made by $\boldsymbol{H}$ to the G and H oracles. (This includes queries made directly by $\boldsymbol{H}$ and those made indirectly as a consequence of $\boldsymbol{H}$'s queries to its $\mathsf{AD}^{G,H}((q,g,x),\cdot)$ oracle.) Since SS is assumed IND-CCA secure and the CDH problem is hard for CG, the advantage functions related to $\boldsymbol{S}$ and $\boldsymbol{C}$ above are negligible, and thus so is the advantage function related to $\boldsymbol{H}$. To complete the proof, we need to specify the adversaries $\boldsymbol{S}, \boldsymbol{C}$ and prove Equation (7).

Description of $\boldsymbol{S}$. Adversary $\boldsymbol{S}$ is given input 1^k and has access to oracles $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{SD}^{R_s}(K,\cdot)$, and R_s. Its goal is to guess the bit b. It begins with the following initializations.

$$((q,g,X),(q,g,x)) \xleftarrow{\$} \mathsf{AK}(1^k)\,;\; y \xleftarrow{\$} \mathbb{Z}_q\,;\; Y \leftarrow g^y\,;\; W \xleftarrow{\$} \{0,1\}^k\,;\; C_a \leftarrow (Y,W)\,;$$
$$T_0 \xleftarrow{\$} \{0,1\}^k\,;\; T_1 \xleftarrow{\$} \{0,1\}^k - \{T_0\}.$$

Subroutine GSim(Z)
    If GT$[Z]$ is not defined then GT$[Z] \stackrel{\$}{\leftarrow} \{0,1\}^k$ EndIf
    Return GT$[Z]$

Subroutine HSim(L)
    If HT$[L]$ is defined then return it as the answer EndIf
    $(\mathrm{dec}, \mathrm{gs}) \stackrel{\$}{\leftarrow} \mathsf{KeyTest}(L)$; $\mathrm{HT}[L] \stackrel{\$}{\leftarrow} \mathbb{Z}_q$
    If dec $= 0$ then return HT$[L]$ as the answer EndIf
    If dec $= 1$ then output gs (as a guess to the value of challenge bit b) and halt EndIf

---

Subroutine $\mathsf{KeyTest}(L)$
    dec $\leftarrow 1$
    For $i = 1, \ldots, k$ do
        $C_0^i[L] \stackrel{\$}{\leftarrow} \mathsf{SE}^{R_s}(K, \mathrm{LR}(T_0, T_0, b))$; If $\mathsf{SD}^{R_s}(L, C_0^i[L]) \neq T_0$ then dec $\leftarrow 0$ EndIf
        $C_1^i[L] \stackrel{\$}{\leftarrow} \mathsf{SE}^{R_s}(K, \mathrm{LR}(T_1, T_1, b))$; If $\mathsf{SD}^{R_s}(L, C_1^i[L]) \neq T_1$ then dec $\leftarrow 0$ EndIf
    EndFor
    If dec $= 0$ return $(0,0)$ EndIf
    For $i = 1, \ldots, k$ do $C^i[L] \stackrel{\$}{\leftarrow} \mathsf{SE}^{R_s}(K, \mathrm{LR}(T_0, T_1, b))$; $T^i \leftarrow \mathsf{SD}^{R_s}(L, C^i[L])$ EndFor
    If $T^1 = T^2 = \cdots = T^k = T_0$ then return $(1,0)$ EndIf
    If $T^1 = T^2 = \cdots = T^k = T_1$ then return $(1,1)$ EndIf
    Return $(0,0)$

Figure 4: Subroutines defined by $\boldsymbol{S}$ and used to simulate $\boldsymbol{H}$'s oracles.

Then it runs $\boldsymbol{H}$ on inputs public key (q, g, X) and ciphertext C_a. In the process $\boldsymbol{H}$ will query its oracles

$$G,\ H,\ R_s,\ \mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b)),\ \mathsf{SD}^{R_s}(K,\cdot),\ \mathsf{AD}^{G,H}((q,g,x),\cdot)\ . \tag{8}$$

$\boldsymbol{S}$ will answer these queries. To that end, it defines the subroutines shown in Figure 4. It answers a query Z to G by running GSim(Z) and returning the answer to $\boldsymbol{H}$. It answers a query L to H by running HSim(L) and returning the answer to $\boldsymbol{H}$. It answers queries to the $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ and R_s oracles via its own oracles of the same name. It answers each query C to the $\mathsf{SD}^{R_s}(K,\cdot)$ oracle using its own decryption oracle, unless there exist i, j and L such that L was queried to H and either $C = C_j^i[L]$ or $C = C^i[L]$. In that case, $\boldsymbol{S}$ aborts. Since $\boldsymbol{S}$ possesses the secret key (q, g, x), it can answer queries to $\mathsf{AD}^{G,H}((q,g,x),\cdot)$ by performing the computation of the decryption algorithm, replacing calls that the latter makes to G or H by calls to the relevant subroutines just mentioned. If $\boldsymbol{H}$ runs to completion ($\boldsymbol{S}$ can output its guess as to the value of b, and halt, before this), then $\boldsymbol{S}$ outputs whatever $\boldsymbol{H}$ outputs.

Description of $\boldsymbol{C}$. Adversary $\boldsymbol{C}$ is given inputs q, g, X, Y, where $X, Y \in \langle g \rangle$ have been chosen uniformly at random. Its goal is to compute g^{xy} where $g^x = X$ and $g^y = Y$. Let $k \leftarrow |\langle 2q+1 \rangle|$. $\boldsymbol{C}$ begins with the following initializations.

$$K \stackrel{\$}{\leftarrow} \mathsf{SK}(1^k)\ ;\ b \stackrel{\$}{\leftarrow} \{0,1\}\ ;\ W \stackrel{\$}{\leftarrow} \{0,1\}^k\ ;\ C_a \leftarrow (Y, W)\ .$$

Then it runs $\boldsymbol{H}$ on inputs public key (q, g, X) and ciphertext C_a. In the process $\boldsymbol{H}$ will query the oracles listed in Equation (8). $\boldsymbol{C}$ will answer these queries. Queries to R_s are simulated the standard way, by returning a random value for each new query and the previously returned value for each repeated query. To simulate the rest of the oracles it defines the subroutines shown in

Subroutine $\mathrm{GSim}(Z)$
  If $\mathrm{GT}[Z]$ is not defined then $\mathrm{GT}[Z] \stackrel{\$}{\leftarrow} \{0,1\}^k$ EndIf
  Return $\mathrm{GT}[Z]$

Subroutine $\mathrm{HSim}(L)$
  If $\mathrm{HT}[L]$ is not defined then $\mathrm{HT}[L] \stackrel{\$}{\leftarrow} \mathbb{Z}_q$ EndIf
  Return $\mathrm{HT}[L]$

Subroutine $\mathrm{ADSim}(Y', W')$
  If there is no L such that $g^{\mathrm{HT}[L]} = Y'$ then return $\perp$ EndIf
  Let L be such that $g^{\mathrm{HT}[L]} = Y'$
  $Z' \leftarrow X^{\mathrm{HT}[L]}$; $T' \leftarrow \mathrm{GSim}(Z')$; $K' \leftarrow T' \oplus W'$; Return K'

Figure 5: Subroutines defined by $\boldsymbol{C}$ and used to simulate $\boldsymbol{H}$'s oracles.

Figure 5. It answers a query Z to G by running $\mathrm{GSim}(Z)$ and returning the answer to $\boldsymbol{H}$. It answers a query L to H by running $\mathrm{HSim}(L)$ and returning the answer to $\boldsymbol{H}$. Since it possesses K and b, it can answer queries to the $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ or $\mathsf{SD}^{R_s}(K,\cdot)$ oracles by simply performing the relevant computation and returning the answer. It answers a query (Y', W') to $\mathsf{AD}^{G,H}((q,g,x),\cdot)$ by running $\mathrm{ADSim}(Y', W')$ and returning the answer. When $\boldsymbol{H}$ has terminated, $\boldsymbol{C}$ picks Z at random from the set $\{Z : \mathrm{GT}[Z] \text{ is defined}\}$ and outputs Z.

Analysis. For the analysis, define the following experiment.

$$\begin{array}{rcl}\mathbf{Exp}^{\mathrm{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) & : & (q,g) \stackrel{\$}{\leftarrow} \mathsf{CG}(1^k)\,;\; x,y \stackrel{\$}{\leftarrow} \mathbb{Z}_q\,;\; Z \leftarrow \boldsymbol{C}(q,g,g^x,g^y) \\ & & \text{If } Z = g^{xy} \text{ then return 1 else return 0}\end{array}$$

We let $\Pr_{\boldsymbol{C}}[\,\cdot\,]$, $\Pr_{\boldsymbol{S}}[\,\cdot\,]$, and $\Pr_{\boldsymbol{H}}[\,\cdot\,]$ denote the probabilities in experiments $\mathbf{Exp}^{\mathrm{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k)$, $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k)$, and $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k)$, respectively.

Let $((q,g,X),(q,g,x)) \in [\mathsf{AK}(1^k)]$ and $K \in [\mathsf{SK}(1^k)]$. We define the following events relating to $\boldsymbol{H}$'s execution on inputs public key (q,g,X) and ciphertext $C_a = (Y, W)$ where $g^y = Y$. These events are defined in any of the three experiments we are considering.

GH	:	There exists a time at which g^{xy} is queried to G but K has not been queried to H
HG	:	There exists a time at which K has been queried to H but g^{xy} has not been queried to G
$\mathsf{Succ}(\boldsymbol{H})$	:	$\boldsymbol{H}$ is successful, meaning its output equals the challenge bit b.

We clarify that the queries referred to above include both direct and indirect queries of $\boldsymbol{H}$, but, in the case of $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k)$, they do *not* include the queries to G and H made by the computation $C_a \leftarrow \mathsf{AE}^{G,H}((q,g,X),\cdot)$ that initializes the experiment. (We are only considering queries to G, H resulting from the execution of $\boldsymbol{H}$.) The main claims related to the analysis are:

$$\Pr_{\boldsymbol{H}}[\,\mathsf{HG} \vee (\mathsf{Succ}(\boldsymbol{H}) \wedge \neg\mathsf{HG} \wedge \neg\mathsf{GH})\,] \;\leq\; \Pr_{\boldsymbol{S}}\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) = 1\,\right] + \frac{O(Q(k))}{2^k} \tag{9}$$

$$\Pr_{\boldsymbol{H}}[\,\mathsf{GH}\,] \;\leq\; Q(k)\cdot\Pr_{\boldsymbol{C}}\left[\,\mathbf{Exp}^{\mathrm{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) = 1\,\right] + \frac{O(Q(k)^2)}{2^k}\,. \tag{10}$$

Let us see how these enable us to conclude the proof, and then return to prove them. We have:

$$\frac{1}{2}\cdot\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) + \frac{1}{2}$$

$$
\begin{aligned}
&= \Pr_{\boldsymbol{H}}\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) = 1\,\right] \\
&= \Pr_{\boldsymbol{H}}[\,\mathsf{Succ}(\boldsymbol{H})\,] \\
&= \Pr_{\boldsymbol{H}}[\,(\mathsf{Succ}(\boldsymbol{H}) \wedge \mathsf{HG}) \vee (\mathsf{Succ}(\boldsymbol{H}) \wedge \neg\mathsf{HG} \wedge \neg\mathsf{GH})\,] + \Pr_{\boldsymbol{H}}[\,\mathsf{Succ}(\boldsymbol{H}) \wedge \mathsf{GH}\,] \\
&\le \Pr_{\boldsymbol{H}}[\,\mathsf{HG} \vee (\mathsf{Succ}(\boldsymbol{H}) \wedge \neg\mathsf{HG} \wedge \neg\mathsf{GH})\,] + \Pr_{\boldsymbol{H}}[\,\mathsf{GH}\,] \\
&\le \Pr_{\boldsymbol{S}}\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) = 1\,\right] + \frac{O(Q(k))}{2^k} + Q(k)\cdot\Pr_{\boldsymbol{C}}\left[\,\mathbf{Exp}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) = 1\,\right] + \frac{O(Q(k)^2)}{2^k} \\
&= \frac{1}{2}\cdot\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) + \frac{1}{2} + Q(k)\cdot\mathsf{Adv}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) + \frac{O(Q(k)^2)}{2^k}\,.
\end{aligned}
$$

Re-arranging terms and simplifying, we get Equation (7). To complete the proof, we must establish Equations (9) and (10).

Proof of Equation (9). An important ingredient in this proof is the following lemma that characterizes what Subroutine $\mathsf{KeyTest}$ accomplishes.

Lemma A.1 If $L = K$ then $\mathsf{KeyTest}(L)$ returns $(1, b)$, while if $L \neq K$ then

$$\Pr\left[\,(\text{dec},\text{gs}) \stackrel{\$}{\leftarrow} \mathsf{KeyTest}(L) \;:\; (\text{dec},\text{gs}) = (1, 1-b)\,\right] \;\le\; 4^{-k}\,. \quad \blacksquare$$

In other words, if $L \neq K$, then with high probability either the test indicates this by returning $\text{dec} = 0$ or it successfully computes the value of the challenge bit b. Above, the probability is over the coin tosses made by the $\mathsf{SE}^{R_s}(K, \text{LR}(\cdot,\cdot,b))$ oracle called in $\mathsf{KeyTest}$, with K and b fixed.

Proof of Lemma A.1: The fact that $\mathsf{KeyTest}(L)$ returns $(1, b)$ when $L = K$ is a consequence merely of the unique decrytability of SS, namely the fact that for all $K \in [\mathsf{SK}(1^k)]$ and all $M \in \{0,1\}^*$ we have $\mathsf{SD}^{R_s}(K, \mathsf{SE}^{R_s}(K, M)) = M$ with probability one, the probability being over the coin tosses of SE.

Now assume $L \neq K$. Let $\Pr[\,\cdot\,]$ denote the probability taken over the coin tosses of $\mathsf{SE}^{R_s}(K,\cdot)$, with K fixed. Let

$$
\begin{aligned}
P_0 &= \Pr\left[\,\mathsf{SD}^{R_s}(L, \mathsf{SE}^{R_s}(K, T_0)) = T_0\,\right] \quad \text{and} \\
P_1 &= \Pr\left[\,\mathsf{SD}^{R_s}(L, \mathsf{SE}^{R_s}(K, T_1)) = T_1\,\right]\,.
\end{aligned}
$$

The probability that $\text{dec} = 1$ at the end of the first For loop in subroutine $\mathsf{KeyTest}$ is $P_0^k P_1^k$ and the probability that $T1 = \cdots = T^k = T_{1-b}$ is at most $(1 - P_b)^k$. So we have

$$
\begin{aligned}
\Pr\left[\,(\text{dec},\text{gs}) \stackrel{\$}{\leftarrow} \mathsf{KeyTest}(L) \;:\; (\text{dec},\text{gs}) = (1, 1-b)\,\right] &= P_0^k P_1^k \cdot (1 - P_b)^k \\
&\le P_b^k \cdot (1 - P_b)^k \\
&= [P_b(1 - P_b)]^k \\
&\le 4^{-k}\,.
\end{aligned}
$$

The last line is true because the function $f\colon [0,1] \to \mathbb{R}$ defined by $f(x) = x(1-x)$ attains its maximum at $x = 1/2$ and the value of this maximum is $1/4$. This concludes the proof. $\blacksquare$

Returning to the proof of Equation (9), we define the following events in $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k)$.

- $\mathsf{FailTest}$: There exists $L \neq K$ such that L was queried to H and $\mathsf{KeyTest}(L)$ returned $(1, 1-b)$ in subroutine $\text{HSim}(L)$
- $\mathsf{Illegit}$: There exist i, j and L such that L was queried to H and either $C^i_j[L]$ or $C^i[L]$ was queried by $\boldsymbol{H}$ to $\mathsf{SD}^{R_s}(K,\cdot)$.

We obtain Equation (9) as shown below. Justifications follow the formulas.

$$
\begin{aligned}
&\Pr_{\boldsymbol{H}}[\,\mathsf{HG} \vee (\mathsf{Succ}(\boldsymbol{H}) \wedge \neg\mathsf{HG} \wedge \neg\mathsf{GH})\,] \\
&\quad\leq\ \Pr_{\boldsymbol{S}}[\,\mathsf{HG} \vee (\mathsf{Succ}(\boldsymbol{H}) \wedge \neg\mathsf{HG} \wedge \neg\mathsf{GH}) \mid \neg\mathsf{FailTest}\,] + \Pr_{\boldsymbol{S}}[\,\mathsf{FailTest}\,] && (11)\\
&\quad\leq\ \Pr_{\boldsymbol{S}}\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) = 1\,\right] + \Pr_{\boldsymbol{S}}[\,\mathsf{Illegit}\,] + \Pr_{\boldsymbol{S}}[\,\mathsf{FailTest}\,] && (12)\\
&\quad\leq\ \Pr_{\boldsymbol{S}}\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) = 1\,\right] + \Pr_{\boldsymbol{S}}[\,\mathsf{Illegit} \mid \neg\mathsf{FailTest}\,] + 2\cdot\Pr_{\boldsymbol{S}}[\,\mathsf{FailTest}\,] \\
&\quad\leq\ \Pr_{\boldsymbol{S}}\left[\,\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) = 1\,\right] + \frac{O(Q(k))}{2^k}\,. && (13)
\end{aligned}
$$

To justify Equation (11), observe that if event $\mathsf{FailTest}$ does not happen, then the simulation of $\boldsymbol{H}$ done by $\boldsymbol{S}$ is correct. (If HG occurs, then prior to this g^{xy} was not a query to G, so the simulation of the G oracle is correct. If $\neg\mathsf{HG} \wedge \neg\mathsf{GH}$ occurs, then also g^{xy} was not a query to G, so the simulation of the G oracle is correct. If $\mathsf{FailTest}$ does not occur, then the replies to queries to H are correct.)

To justify Equation (12), first note that if event HG occurs, then the $L = K$ case of Lemma A.1 tells us that $\boldsymbol{S}$ halts with correct output. On the other hand, if neither HG nor GH occur, then $\boldsymbol{S}$ halts with correct output as long as $\boldsymbol{H}$ does. But $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k)$ can still fail to return 1 because $\boldsymbol{S}$ aborted due to the occurrence of $\mathsf{Illegit}$. (When the latter occurs, $\boldsymbol{S}$ aborts to avoid calling its oracle $\mathsf{SD}^{R_s}(K,\cdot)$ on a ciphertext returned by its $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ oracle.)

To justify Equation (13), first note that Lemma A.1 together with the fact that the total number of queries is at most $Q(k)$ implies that $\Pr_{\boldsymbol{S}}[\,\mathsf{FailTest}\,] \leq Q(k)/4^k$. Next, we observe that if $\mathsf{FailTest}$ does not occur, then $\boldsymbol{H}$ gets no information about T_0, T_1 other than that they are random distinct k-bit strings. The unique decryptability of SS then tells us that $\Pr_{\boldsymbol{S}}[\,\mathsf{Illegit} \mid \neg\mathsf{FailTest}\,]$ is bounded above by the probability of guessing either T_0 or T_1 in $Q(k)$ tries, and this is $O(Q(k)/2^k)$.

Proof of Equation (10). We define the following event in $\mathbf{Exp}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k)$.

$\mathsf{FailDec}$: There exist times $t' < t$ and Y', W', L such that all the following hold:
- query (Y', W') was made to $\mathsf{AD}^{G,H}((q,g,x),\cdot)$ at time t' and $\mathrm{ADSim}(Y', W')$ returned $\perp$
- query L was made to H at time t
- $g^{\mathrm{HT}[L]} = Y'$.

The answers provided by $\mathrm{ADSim}(\cdot,\cdot)$ are correct exactly when this event does not occur. Furthermore, if there is a time at which query g^{xy} to G occurs and GH is true, then query K to H has not occurred at this time, and thus the answers to queries to H have been correct. Hence

$$
\Pr_{\boldsymbol{C}}\left[\,\mathbf{Exp}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) = 1\,\right] \;\geq\; \frac{\Pr_{\boldsymbol{H}}[\,\mathsf{GH}\,] - \Pr_{\boldsymbol{C}}[\,\mathsf{FailDec}\,]}{Q(k)}\,.
$$

Re-arranging, we get

$$
\Pr_{\boldsymbol{H}}[\,\mathsf{GH}\,] \;\leq\; Q(k)\cdot\Pr_{\boldsymbol{C}}\left[\,\mathbf{Exp}^{\text{cdh}}_{\mathsf{CG},\boldsymbol{C}}(k) = 1\,\right] + \Pr_{\boldsymbol{C}}[\,\mathsf{FailDec}\,]\,. \tag{14}
$$

At any point in time, a query L to H has probability at most ℓ/q of making $\mathsf{FailDec}$ happen, where ℓ is the number of queries that have been made to $\mathsf{AD}^{G,H}((q,g,x),\cdot)$ at this time. Recall that $k = |\langle 2q+1\rangle|$ and thus $q \geq 2^{k-2}$. Putting these observations together we get

$$
\Pr_{\boldsymbol{C}}[\,\mathsf{FailDec}\,] \;\leq\; \frac{Q(k)^2}{q} \;\leq\; \frac{Q(k)2}{2^{k-2}} \;=\; \frac{O(Q(k)^2)}{2^k}\,.
$$

Putting this together with Equation (14) completes the proof of Equation (10).

B Any IND-CCA-secure scheme is IND-CCA preserving

We remarked in Section 1.2 that any asymmetric encryption scheme that is IND-CCA secure is also IND-CCA preserving. (The interesting thing about the Hash ElGamal scheme is that it is not IND-CCA secure but is still IND-CCA preserving.) For completeness, we state and and prove this formally here. We begin by recalling the definition of IND-CCA security of an asymmetric encryption scheme.

Definition. This follows [4]. Associate to AS, an adversary $\boldsymbol{A}$, and $k \in \mathbb{N}$, the following experiment.

Experiment $\mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}}(k)$
  Randomly choose RO $R_a\colon \{0,1\}^* \to \{0,1\}$
  $(pk, sk) \xleftarrow{\$} \mathsf{AK}^{R_a}(1^k)$; $b \xleftarrow{\$} \{0,1\}$
  Run $\boldsymbol{A}$ with input $1^k, pk$ and oracles $\mathsf{AE}^{R_a}(pk, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{AD}^{R_a}(sk,\cdot)$, R_a
  Let d denote the output of $\boldsymbol{A}$
  If $d = b$ then return 1 else return 0.

We say that adversary $\boldsymbol{A}$ is legitimate if it never queries $\mathsf{AD}^{R_a}(sk,\cdot)$ with a ciphertext previously returned by $\mathsf{AE}^{R_s}(pk, \mathrm{LR}(\cdot,\cdot,b))$. Asymmetric encryption scheme AS is said to be IND-CCA secure if the function

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}}(k) \;=\; 2 \cdot \Pr\left[\, \mathbf{Exp}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}}(k) = 1 \,\right] - 1$$

is negligible for all legitimate polynomial-time adversaries $\boldsymbol{A}$. IND-CPA security is defined similarly, except the adversary is not given access to oracle $\mathsf{AD}^{R_a}(sk,\cdot)$.

Result. The following holds in both the standard and the RO models.

Theorem B.1 Let AS be an IND-CCA-secure asymmetric encryption scheme. Then AS is IND-CCA preserving. ▮

Proof of Theorem B.1: Let $\mathsf{AS} = (\mathsf{AK}, \mathsf{AE}, \mathsf{AD})$ be an IND-CCA-secure asymmetric encryption scheme and let $\mathsf{SS} = (\mathsf{SK}, \mathsf{SE}, \mathsf{SD})$ be an IND-CCA-secure symmetric encryption scheme. We will show that for any polynomial-time legitimate hybrid adversary $\boldsymbol{H}$ attacking mm-hybrid encryption scheme $(\mathsf{AS}, \mathsf{SS})$ there exist polynomial-time legitimate adversaries $\boldsymbol{A}$ and $\boldsymbol{S}$ such that for any $k \in \mathbb{N}$

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) \;\leq\; 2\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}}(k) + \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k) \;. \tag{15}$$

Since AS and SS are assumed IND-CCA secure, the advantage functions related to $\boldsymbol{A}$ and $\boldsymbol{S}$ above are negligible, and thus so is the advantage function related to $\boldsymbol{H}$. To complete the proof, we need to specify the adversaries $\boldsymbol{A}, \boldsymbol{S}$ and prove Equation (15).

We first associate to $(\mathsf{AS}, \mathsf{SS})$, $\boldsymbol{H}$, and $k \in \mathbb{N}$, the following experiments, for $i \in \{1,2,3,4\}$.

Experiment $\mathbf{Exp}^{i}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k)$
  Randomly choose RO $R\colon \{0,1\}^* \to \{0,1\}$
  Define ROs $R_s(\cdot) = R(0\|\cdot)$ and $R_a(\cdot) = R(1\|\cdot)$
  $(pk, sk) \xleftarrow{\$} \mathsf{AK}^{R_a}(1^k)$; $K \xleftarrow{\$} \mathsf{SK}^{R_s}(1^k)$; $K' \xleftarrow{\$} \mathsf{SK}^{R_s}(1^k)$
  If $i = 1$ or $i = 4$ then $C_a \xleftarrow{\$} \mathsf{AE}^{R_a}(pk, K)$ else $C_a \xleftarrow{\$} \mathsf{AE}^{R_a}(pk, K')$ EndIf
  If $i = 1$ or $i = 2$ then run $\boldsymbol{H}$ with inputs pk, C_a and oracles

$\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,0))$, $\mathsf{SD}^{R_s}(K,\cdot)$, $\mathsf{AD}^{R_a}(sk,\cdot)$, R
Else run $\boldsymbol{H}$ with inputs pk, C_a and oracles
$\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,1))$, $\mathsf{SD}^{R_s}(K,\cdot)$, $\mathsf{AD}^{R_a}(sk,\cdot)$, R
EndIf
Let d denote the output of $\boldsymbol{H}$
Return d.

For $i \in \{1,2,3,4\}$, let P_i denote the probability that $\mathbf{Exp}^{i}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k)$ returns 1. It is easy to see that

$$\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\mathsf{SS},\boldsymbol{H}}(k) \;=\; P_4 - P_1 \;=\; (P_4 - P_3) + (P_3 - P_2) + (P_2 - P_1)\;.$$

We will show that there exist legitimate polynomial-time adversaries $\boldsymbol{A}'$, $\boldsymbol{S}$, and $\boldsymbol{A}''$ such that

$$P_4 - P_3 \;=\; \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}'}(k)\,,\quad P_3 - P_2 \;=\; \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{SS},\boldsymbol{S}}(k)\,,\quad \text{and} \quad P_2 - P_1 \;=\; \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}''}(k)\;. \tag{16}$$

We obtain Equation (15) from the above by setting $\boldsymbol{A} = \boldsymbol{A}'$ if $\mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}'}(k) \geq \mathsf{Adv}^{\text{ind-cca}}_{\mathsf{AS},\boldsymbol{A}''}(k)$, and $\boldsymbol{A} = \boldsymbol{A}''$ otherwise. We now define adversaries $\boldsymbol{A}'$, $\boldsymbol{A}''$, $\boldsymbol{S}$ and prove Equation (16).

Description of $\boldsymbol{A}'$. Adversary $\boldsymbol{A}'$ is given inputs $1^k, pk$ and has access to oracles $\mathsf{AE}^{R_a}(pk, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{AD}^{R_a}(sk,\cdot)$, and R_a. Its goal is to guess the bit b. It begins with the following initializations.

$K \stackrel{\$}{\leftarrow} \{0,1\}^k$; $K' \stackrel{\$}{\leftarrow} \{0,1\}^k$
Make query (K', K) to $\mathsf{AE}^{R_a}(pk, \mathrm{LR}(\cdot,\cdot,b))$, and let C_a be the response

Then it runs $\boldsymbol{H}$ on inputs public key pk and ciphertext C_a. In the process $\boldsymbol{H}$ will query its oracles

$$R_a,\ R_s,\ \mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b)),\ \mathsf{SD}^{R_s}(K,\cdot),\ \mathsf{AD}^{R_a}(sk,\cdot)\;. \tag{17}$$

$\boldsymbol{A}'$ will answer these queries. Queries to R_s are simulated the standard way, by returning a random value for each new query and the previously returned value for each repeated query. $\boldsymbol{A}'$ answers queries to the $\mathsf{AD}^{R_a}(sk,\cdot)$ and R_a oracles via its own oracles of the same name. Since it possesses K, it can answer queries to $\mathsf{SD}^{R_s}(K,\cdot)$ by simply performing the computation of the decryption algorithm, replacing calls that the latter makes to R_s by the above-mentioned simulation, and returning the answer. $\boldsymbol{A}'$ answers queries to $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ by using K to simulate oracle $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,1))$. When $\boldsymbol{H}$ halts and outputs d, $\boldsymbol{A}'$ outputs d.

Description of $\boldsymbol{A}''$. Adversary $\boldsymbol{A}''$ is identical to adversary $\boldsymbol{A}'$, except that it makes query (K, K') to oracle $\mathsf{AE}^{R_a}(pk, \mathrm{LR}(\cdot,\cdot,b))$ and it answers queries to $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$ by using K to simulate oracle $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,0))$.

Description of $\boldsymbol{S}$. Adversary $\boldsymbol{S}$ is given input 1^k and has access to oracles $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{SD}^{R_s}(K,\cdot)$, and R_s. Its goal is to guess the bit b. It begins with the following initializations.

$K' \stackrel{\$}{\leftarrow} \{0,1\}^k$; $(pk, sk) \stackrel{\$}{\leftarrow} \mathsf{AK}^{R_a}(1^k)$; $C_a \stackrel{\$}{\leftarrow} \mathsf{AE}_{pk}(K')$

Then it runs $\boldsymbol{H}$ on inputs public key pk and ciphertext C_a. In the process $\boldsymbol{H}$ will query the oracles listed in Equation (17). $\boldsymbol{S}$ will answer these queries. Queries to R_a are simulated the standard way, by returning a random value for each new query and the previously returned value for each repeated query. $\boldsymbol{S}$ answers queries to the $\mathsf{SE}^{R_s}(K, \mathrm{LR}(\cdot,\cdot,b))$, $\mathsf{SD}^{R_s}(K,\cdot)$, and R_s oracles via its own oracles of the same name. Since it possesses the secret key sk, it can answer queries to $\mathsf{AD}^{R_a}(sk,\cdot)$

by simply performing the computation of the decryption algorithm, replacing calls that the latter makes to R_a by the above-mentioned simulation, and returning the answer. When $\boldsymbol{H}$ halts and outputs d, $\boldsymbol{S}$ outputs d.

Analysis. Clearly, if $\boldsymbol{H}$ is polynomial-time and legitimate, so are $\boldsymbol{A}'$, $\boldsymbol{A}''$, and $\boldsymbol{S}$. It is easy to see that Equation (16) holds. ∎