On the Hardness of Proving CCA-Security of Signed ElGamal

David Bernhard[1] and Marc Fischlin[2] and Bogdan Warinschi[1]

[1]University of Bristol [2]Technische Universität Darmstadt

Abstract. The well-known Signed ElGamal scheme consists of ElGamal encryption with a non-interactive Schnorr proof of knowledge. While this scheme should be intuitively secure against chosen-ciphertext attacks in the random oracle model, its security has not yet been proven nor disproven so far, without relying on further non-standard assumptions like the generic group model. Currently, the best known positive result is that Signed ElGamal is non-malleable under chosen-plaintext attacks. In this paper we provide evidence that Signed ElGamal may not be CCA secure in the random oracle model. That is, building on previous work of Shoup and Gennaro (Eurocrypt'98), Seurin and Treger (CT-RSA 2013), and Bernhard et al. (PKC 2015), we exclude a large class of potential reductions that could be used to establish CCA security of the scheme.

1 Introduction

Indistinguishability under chosen-ciphertext attacks (IND-CCA, or CCA for short) is widely considered to be the appropriate security notion for public-key encryption. Most known CCA-secure public-key schemes are built from a basic IND-CPA scheme (like ElGamal) and a non-interactive proof system. Examples of this approach include Cramer-Shoup [7], TDH2 [16], and the Chaum-Pedersen-Signed ElGamal scheme of Seurin and Treger [14], but also more theoretical constructions like the DDN encryption scheme [8] fall under this paradigm.

The difference between IND-CPA and IND-CCA security is that the latter notion allows the adversary to see decryptions of ciphertexts via a decryption oracle. Informally, encrypt-then-prove schemes require an adversary to prove knowledge of a plaintext as part of a valid ciphertext. But then a decryption oracle which the adversary can only call for ciphertexts on messages that she already knows is, intuitively, redundant. Hence, the encrypt-then-prove should reduce CCA-security to IND-CPA of the basic scheme. Interestingly, this intuition appears to be hard to turn into a formal proof, as we discuss for the case of the Signed ElGamal encryption scheme.

1.1 Signed ElGamal

Signed ElGamal [16,13] is a well-known "encrypt-then-prove" scheme combining ElGamal encryption with a Fiat-Shamir-Schnorr proof [12,9] of the randomness used to encrypt (from which you can recover the plaintext from a ciphertext). It is the most efficient encrypt-then-prove scheme known to date both in terms of ciphertext size and computation cost. Further, it is submission-secure [19] (i.e., one can work homomorphically with the "core" ElGamal ciphertexts) and publicly verifiable, making it suitable for applications such as electronic voting[1]. All these properties would make Signed ElGamal the "primary choice" for a CCA encryption scheme, unless one objects to the Random Oracle Model (ROM) [3] methodology. Remarkably, however, Signed ElGamal has never been proven to be CCA-secure, even in the Random Oracle Model!

Shoup and Gennaro [16] were the first to consider the security of Signed ElGamal. They found that the "obvious" proof strategy to show CCA-security based on the encrypt-then-prove intuition did not work and gave a concrete example why the common strategy fails, but neither proved nor disproved CCA-security of the scheme. Instead, they developed the slightly less efficient TDH2 scheme which does come with a CCA proof. Schnorr and Jakobsson [13] proved Signed ElGamal to be CCA-secure under a combination of the ROM and the generic group model (GGM). Tsiounis and Yung [18] gave yet another proof under a non-standard "knowledge assumption" that resembles the approach behind the GGM.

More abstractly, the two mentioned proofs of CCA-security of Signed ElGamal [18,13] both rely on variants of a property known as plaintext awareness, which together with IND-CPA security suffices to show CCA; this property is in fact strictly stronger than CCA-security [4]. Plaintext awareness requires the existence of a plaintext extractor who, given some trapdoor key, can extract plaintexts from ciphertexts in an "online" manner, i.e., without interacting further with the party who created the ciphertext. However, in 2013, Seurin and Treger [14] showed that a plaintext extractor for Signed ElGamal in the ROM, without extra assumptions such as the GGM, could not exist, unless plain ElGamal is already insecure (i.e. one can solve the Computational Diffie-Hellman (CDH) problem in the underlying group). This result calls into question the proofs based on plaintext awareness as a route to show CCA-security of Signed ElGamal.

We stress again that Seurin and Treger, like Shoup and Gennaro, did not prove or disprove the CCA-security of Signed ElGamal in the ROM. Their result only rules out proofs based on plaintext awareness. Also, the recent result of Bernhard et al. [5], showing that Fiat-Shamir-Schnorr proofs of knowledge are not adaptively secure, only gives a limited answer about the CCA-security of Signed ElGamal. Their result relies on the fact the the knowledge extractor has to return the full witness (i.e., the randomness for Signed ElGamal), whereas a clever reduction only needs to simulate the decryption oracle, returning the message as a fraction of the full witness.

In this paper we provide further evidence against the CCA security of Signed ElGamal, even if one takes a direct route, without going through plaintext aware-

[1] The Helios [2] voting scheme used by the IACR uses a variant of Signed ElGamal.

ness. To this end, we rule out a large class of common proof techniques. The obstacle encountered by Shoup and Gennaro seems to be very solid indeed.

1.2 State of the art

We summarise previous work on the security of Signed ElGamal in the following table, ordered by the strength of the model.

paper	**result**	**model**
[6]	NM-CPA secure	ROM
[16]	obvious CCA proof fails	ROM
[14]	not PA2 unless CDH easy	ROM
[5]	no adaptive extractor under OMDL	ROM
this work	**no CCA reduction under IES**	**ROM**
[1]	CCA	ROM vs. algebraic adv.
[13,18]	CCA	ROM+GGM

Although CCA security of Signed ElGamal has sometimes been claimed informally, the strongest formal result in the ROM to date [6] only shows the weaker non-malleability (NM-CPA). If one extends the ROM to include either the generic group model [13], a generic knowledge assumption [18] or restricts to algebraic adversaries [1] then one can prove CCA security. Conversely, we know that in the plain ROM the "obvious" CCA proof fails [16]. Signed ElGamal cannot be ROM-PA2 plaintext aware unless CDH is easy [14] which rules out proofs based on non-rewinding extractors for the contained ZK proof. The strongest negative result to date [5] rules out any CCA proof based on adaptive extractors for the ZK proof, by showing that the proof scheme in question (Fiat-Shamir-Schnorr) is not adaptively secure unless the OMDL problem is easy.

Adaptive proofs of knowledge. To put our results in context, we outline and discuss the results of Bernhard et al. [5], explain their limits and how we improve on them. Their work identifies the notion of adaptive proofs of knowledge as a potential bottleneck towards proving IND-CCA security for Signed ElGamal: this notion is what seems to be needed to make the intuition behind encypt-then-prove work, yet it is provably not achieved by its implementation based on Fiat-Shamir-Schnorr proofs in Signed ElGamal.

In a hypothetical CCA-to-IND-CPA reduction of Signed to plain ElGamal, when the adversary asks a decryption query on a ciphertext, the reduction rewinds the adversary to extract the plaintext. Shoup and Gennaro [16] considered an adversary who makes a chain of n ciphertexts, the plaintext of each one depending on the last ciphertext (e.g. through a hash function), then asks decryption queries in reverse order. The effect of this adversary is to make a straightforward rewinding strategy take exponential time in n, as the reduction ends up re-rewinding the rewound adversaries each time.

Bernhard et al. [5] were the first to show that such an exponential expansion is unavoidable under certain conditions. A non-interactive proof of knowledge is, informally speaking, a construction in which you can extract a witness from a single proof given suitable powers (e.g. the ability to rewind the prover). Bernhard et al. proposed a notion of adaptive proofs in which the prover can make a sequence of proofs and the extractor must return the found witnesses to the prover. In this way, should the prover ever succeed in making a proof for which she does not know a witness, she gains knowledge from the extractor. The game is adaptive in the sense that the extractor must deliver the witness for the k-th proof to the prover before the prover makes proof $k + 1$.

Bernhard et al. proved that (unlike a construction of Fischlin [10]) Fiat-Shamir-Schnorr proofs are not adaptively secure, unless the one-more discrete logarithm (OMDL) problem is easy in the group concerned. Specifically, any adaptive extractor must either take at least 2^n time on an adapted version of Shoup/Gennaro's adversary, or reduce to solving OMDL.

Their results rules out a proof of IND-CCA security for Signed ElGamal which considers the basic encryption scheme and the non-interactive proofs in isolation, such as one might do following the encrypt-then-prove intuition. The intuition is that the reduction would not be able to answer decryption queries by relying on an extractor for the Fiat-Shamir-Schnorr proofs, as such an extractor does not exists.

Strictly speaking however, their result only rules out an extractor that obtains the randomness used in the ciphertexts and not one that somehow obtains only the underlying plaintexts. Yet, there is a significant complexity gap between these two problems: finding the plaintext is equivalent to solving a CDH problem (with the aid of rewinding) whereas finding the randomness (again with the aid of rewinding) is equivalent to taking a discrete logarithm. This means that the result outlined above does not rule out all plausible reductions. In addition, the result does not immediately apply to the combination of ElGamal ciphertext and proof that makes up Signed ElGamal.

1.3 Our Contribution

We narrow the gap between the positive and negative results by showing that, in the ROM, one cannot construct any black-box key-preserving reduction from CCA-security of Signed ElGamal to IND-CPA security of plain ElGamal unless Schnorr proofs are insecure (specifically, they can help you solve CDH). We view this result as strong evidence in favour of the hypothesis that Signed ElGamal is *not CCA secure* in the ROM.

Technically, we show a metareduction which turns any reduction from the IND-CCA security to IND-CPA of ElGamal into an adversary against an assumption which we call the "Interactive ElGamal-Schnorr" assumption, or IES in short. Informally, the IES assumption is the following.

> You are given an ElGamal public key and a ciphertext on an unknown, random message. You can play the verifier in a single interactive Schnorr proof of the randomness in the ciphertext. Then you cannot extract the encrypted message.

We remark that we are *not* proposing a new assumption for the purpose of giving a cryptosystem that is secure under this assumption. Instead, we are showing that an already well-known cryptosystem cannot be proven CCA secure unless a plausible assumption is actually *false* — in which case we would be distrustful of any cryptosystem employing Schnorr proofs. Since IES is closely related to CDH, we would also have concerns about the use of any ElGamal-based scheme in a group in which IES is easy.

1.4 Outline of this work

We begin by recalling the definition of Signed ElGamal and the IND-CPA/CCA notions for encryption. We then present and justify the IES assumption and prove that for any group, if there is a subexponential key-passing reduction from CCA of Signed ElGamal to IND-CPA of ElGamal then IES is efficiently breakable in the group concerned.

2 Preliminaries

2.1 Cryptographic groups

A cryptographic group is a group G of some prime order q together with a designated generator g, in which one can perform the group operation and inversion efficiently. It follows that one can also efficiently exponentiate in such groups. Typical examples (that have interesting security properties) are subgroups of the multiplicative group $\mathbb{Z}_p^\times$ for primes p and groups derived from elliptic curves over finite fields.

2.2 Public-key encryption and ElGamal

A public-key encryption scheme consists of three algorithms: KeyGen which produces a public and a secret key, $\mathsf{Encrypt}$ which takes a message and a public key and produces a ciphertext and $\mathsf{Decrypt}$ which takes a secret key and ciphertext and produces either a message or the symbol $\perp$ to indicate failure. Decryption is deterministic. If you generate a key pair, encrypt a message with the public key then decrypt the ciphertext with the matching secret key then you get the same message back.

The ElGamal encryption scheme over a group G (generated by g, of order q) has key pairs of the form (g^x, x) for $x \in \mathbb{Z}_q$; to generate a secret key one picks a random integer x modulo q. To encrypt a message $m \in G$ to public key $y \in G$, pick a random $r \in \mathbb{Z}_q$, your ciphertext is $(g^r, m \cdot y^r)$. To decrypt a ciphertext (c, d) with secret key x compute d/c^x.

The IND-CPA and IND-CCA security notions are given by the following game. To begin, the game generates a key pair and returns the public key. Once in the game, you may pick two messages (m_0, m_1) of the same length[2] in

[2] For ElGamal, the message space is the underlying group G and all group elements have the same length which can be described as "one element".

response to which the game picks $\beta \in \{0, 1\}$ randomly and gives you a challenge encryption c^* of m_β. In the CCA version of the game only, you may ask the game to decrypt any ciphertext for you, as often as you like and both before and after obtaining the challenge — except that after obtaining the challenge c^*, you may not ask for c^* itself to be decrypted. Your aim is to guess β. Your success probability σ is the probability that you guess β correctly (taken over all random choices made by the game) and your advantage α is defined as $2\sigma - 1$, so a perfect guesser has advantage 1 and a uniform random guesser has advantage 0.

For a sequence of groups G_λ indexed by a security parameter $\lambda \in \mathbb{N}$, the ElGamal encryption scheme is said to be (asymptotically) IND-CPA/CCA secure if the advantage of any efficient adversary (who receives λ as input in unary notation) in the corresponding game over group G_λ is negligible as a function of the parameter λ.

ElGamal (a.k.a. plain ElGamal) is IND-CPA secure under the DDH assumption: given a pair (g^x, g^y) of uniformly random and independent group elements it is hard to tell g^{xy} from another independent, uniformly random group element g^z. More precisely, there is a reduction from breaking IND-CPA of ElGamal to solving DDH that succeeds 1/2 of the time. Plain ElGamal is not CCA secure.

2.3 Schnorr proofs

Over a cryptographic group G with designated generator g and order q, the Schnorr proof scheme is a protocol for a prover to convince a verifier that he knows a secret x such that $y = g^x$, where y may be known to the verifier in advance. The prover picks a random $a \in \mathbb{Z}_q$ and sends y and g^a to the verifier who replies with a challenge c drawn randomly from $\mathbb{Z}_q$. The prover answers with $s = a + cx \pmod{q}$ and the verifier accepts if and only if $g^s = g^a \cdot y^c$.

The Fiat-Shamir-Schnorr protocol is a non-interactive version of the above. Instead of the verifier picking c, the prover picks it herself as $c = H(y, g^a)$ where H is a cryptographic hash function with codomain $\mathbb{Z}_q$. The prover sends the verifier a single message (y, g^a, s) and the verifier recomputes[3] c and performs the same check as in the interactive protocol. Fiat-Shamir-Schnorr requires the so-called Random Oracle Model (ROM) for its security analysis, which idealises the hash function as an oracle that both prover and verifier can call.

2.4 Signed ElGamal

Signed ElGamal combines plain ElGamal and a Fiat-Shamir-Schnorr proof in a construction that Fischlin et al. call encrypt-then-prove. We define the scheme formally here.

[3] A variant of the protocol has the prover send (y, c, s) which is often shorter as it consists of one group element and two integers instead of two group elements and one integer. This variant is identical to the protocol presented here for security purposes.

Definition 1. *Signed ElGamal is the following encryption scheme over a cryptographic group G of order q with generator g.*

KeyGen: *Pick $x \in \mathbb{Z}_q$ uniformly at random and set $y = g^x$ for your public key. Your keypair is (y, x).*

Encrypt: *Your message m must be an element of G. Let y be the public key. Pick a random $r \in \mathbb{Z}_q$ and compute an ElGamal ciphertext $(g^r, y^r \cdot m)$. Then make a Fiat-Shamir-Schnorr proof: pick random $a \in \mathbb{Z}_q$, set $c = H(y, g^r, y^r \cdot m, g^a)$ and compute $s = a + cx \pmod{q}$. Your ciphertext is $(g^r, y^r \cdot m, g^a, s)$.*

Decrypt: *Given x and a ciphertext (u, v, b, s) compute $c = H(g^x, u, v, b)$ and check that $g^s = b \cdot u^c$. If this check fails, the ciphertext is invalid — return $\perp$. Otherwise decrypt $m = v/u^x$.*

2.5 Metareductions

A cryptographic security definition often takes the form of a game: an algorithm with one interface and a notion of winning. Specifically, a scheme is secure if there is no efficient adversary (an algorithm with one interface, compatible with that of the game) such that if we connect the adversary to the game, the adversary wins (with more than a negligible chance).

A reduction from source problem (e.g. IND-CPA of ElGamal) to a target problem (e.g. DDH) is an algorithm with two interfaces, one for a source-problem adversary and one for the target-problem game. The aim of a proof by reduction is to show that for any adversary who could win the source game, the system obtained by composing the adversary and the reduction would win the target game. This system is itself an algorithm with one interface, which is compatible with the target game.

A metareduction is an algorithm with three interfaces. A proof by metareduction shows that there can be no reduction from a source problem S to a target problem T unless another problem U is already easy. The metareduction's first two interfaces are those of an S-adversary and a T-game; the third interface is compatible with the U-game. In a proof by metareduction, we take a hypothetical S-to-T reduction and connect its S and T interfaces to those of the metareduction. In other words, composing a S, T, U metareduction with a S, T reduction gives a system with one free interface of type U, and this whole system can be connected to the U-game.

A metareduction will typically simulate a perfect S-adversary. The accompanying proof will show that if the reduction wins the T-game given a perfect S-adversary, then the metareduction wins the U-game given the reduction.

3 The IES assumption

The interactive Schnorr proof scheme is known to be a correct, honest-verifier zero-knowledge proof of knowledge of a discrete logarithm. The non-interactive (Fiat-Shamir-Schnorr) version is "full" zero-knowledge in the ROM. We propose an assumption that we call IES (Interactive ElGamal-Schnorr) that looks at an interactive Schnorr proof on an ElGamal ciphertext for a random message. While weaker than assuming such a proof to be zero-knowledge, IES states the assumption that such a proof does not leak the encrypted message.

Suppose you are given an ElGamal public key $y = g^x$ and an encryption $(u, v) = (g^r, my^r)$ for a random group element m. In addition, you receive a Schnorr commitment g^a for a random a and can pick a $c \in \mathbb{Z}_q$, in response to which you get $s = a + cr \pmod{q}$. The IES assumption is then that you cannot recover m.

It turns out that m is actually not required to state IES. Decrypting an ElGamal ciphertext is solving a CDH[4] instance, so we can state IES as a CDH variant directly:

Definition 2. *Given three uniformly random and independent group elements (g^x, g^r, g^a) in a cryptographic group G of order q with generator g, the IES problem is to compute g^{rx} (the CDH problem) where one, after receiving (g^x, g^r, g^a), may pick a single value $c \in \mathbb{Z}_q$ and learns $s = a + cr \pmod{q}$ as a one-time auxiliary information.*

This definition shows that IES is stronger than CDH since a CDH solver could break IES trivially. The justification that s should not help is the same one as for the interactive Schnorr proof: since a is uniformly random in $\mathbb{Z}_q$ and independent of r, x, if g^a were not provided then s would be uniform and independent of r, x itself and the problem would reduce to CDH. The IES assumption formalises the idea that giving out g^a as well, which is also independent of the CDH problem on r, x, should not help you either.

For IES adversaries who pick c independently of a, the IES assumption reduces to CDH with the help of a rewinding reduction. Given a CDH instance (g^x, g^r) one can pick a random g^a and run the adversary up to the point where she produces c, then pick a random s and set $h = g^s/(g^r)^c$ and rerun the adversary on (g^x, g^r, h). As long as the same c appears in the second run, the simulation is sound (in particular the adversary can verify that she got the correct s). This is of course exactly how one simulates Schnorr proofs to show honest-verifier zero-knowledge of the protocol. Like for Schnorr proofs, the simulation argument breaks down if the adversary chooses c depending on g^a but there is no known attack to exploit this technique.

The difference between breaking IES and extracting a witness from a Schnorr proof is that the former requires only finding a particular group element whereas the latter involves recovering an integer (exponent). An adversary who can recover x from a Schnorr proof $(g^x, g^a, c, s = a + cx)$ can take discrete logarithms.

The result of Bernahrd et al. on Fiat-Shamir-Schnorr shows that one cannot build a CCA–to–IND-CPA reduction for Signed ElGamal by extracting the witness (the encryption randomness) from the Schnorr proof in a ciphertext. However, the main task for such a reduction is to answer decryption queries, for which it suffices to recover the encrypted message (a group element).

[4] Computational Diffie-Hellman: given random group elements g^x, g^y compute g^{xy}.

4 Main theorem

Our goal is to exclude reductions from CCA security of Signed ElGamal to IND-CPA security of plain ElGamal (equivalently, to DDH). We make three constraints on the class of reductions that we consider. First, we consider only efficient reductions, since an exponential-time reduction could exhaustively search the key-space. Secondly, we consider rewinding black-box reductions: our reductions may invoke any number of copies of the adversary as long as the reduction is efficient overall. Each invocation of the adversary counts as a single operation. All these copies of the adversary run with the same random string. The reduction is in charge of all communication to and from these copies, including random oracle calls. In particular the reduction can employ the usual "special soundness" forking strategies. All computation from the moment the reduction sends a message to a copy of the adversary up to the adversary's reply counts as a single operation as far as the reduction is concerned.

Finally, we consider only key-passing reductions. A reduction to IND-CPA receives a public key from the IND-CPA challenger whereas a CCA adversary expects a public key; keys for plain and Signed ElGamal are of the same form. A key-passing reduction is one that gives all copies of the adversary the same public key which it received from its challenger. Alternatively, one could view the public key as being made available globally to all parties (the reduction and the copies of the adversary) via the IND-CPA challenger.

Why key-passing? We provide some intuition and the technical reasons for the key-passing assumption. When deploying public-key encryption, the problem of key authentication has to be solved before you can start to encrypt anything: how do you know that the public key you have really belongs to the person it claims to be from? Otherwise, you may be vulnerable to fake-key attacks in which you accidentally encrypt a secret under an adversary's public key, which they can decrypt immediately.

One solution is to have public keys publicly certified in a directory, a so-called Public Key Infrastructure (PKI). Our model considers an adversary attacking a particular public key, which we presume that our adversary knows from the start of the experiment. This would be the case if a PKI were deployed. Due to the rewinding nature of the Schnorr protocol extractor, we must allow our reduction access to multiple copies of the adversary with full control over the random oracle. Yet we do not want to offer the reduction the option to substitute a key of its own (for which it may know the secret key) for some copies of the adversary, leading to the following fake-key attack: the reduction starts one copy of the adversary with the IND-CPA challenger's public key and a second copy of the adversary with a public key of its own. Whenever the adversary with the "real" key sends a challenge ciphertext, the reduction runs the fake-key adversary up to the same point, then simply decrypts the challenge with its secret key.

Maybe this problem could be solved by having the adversary choose all challenge messages as a hash of some values including the public key, thus two different copies of the adversary run with different keys would also produce different challenges. But this would complicate the proof unnecessarily to deal with a scenario that we are not trying to model in the first place.

Note that it would not be sufficient to ask that each public key comes with a Schnorr signature of knowledge of its secret key (perhaps signed by the challenger) — the reduction controls the random oracle towards the adversary, so it could easily forge such signatures.

Technically, the fake-key problem appears when our metareduction tries to inject an IES challenge into the reduction's view. The reduction, on input the challenger's public key pk, could both substitute a key of its own (for which it knows the secret key, but which is independent of the IES challenger) or the reduction could rerandomise pk by picking random r and returning pk^r. In this case the reduction cannot directly decrypt anything, but there is a dependency on the IES challenger's key. Intuitively, creating further keys of its own should not help the reduction to attack the challenger. But to a metareduction, both these tactics are indistinguishable: the resulting key looks random in both cases. If the metareduction injects an IES challenge into a ciphertext for which the reduction knows the secret key, all bets are off — the metareduction cannot simulate an adversary consistently anymore.

It would be interesting to see whether the key-passing requirement could be weakened in future work. A more complicated argument (with looser concrete security bounds) may well succeed, but for now we prefer to work in the key-passing model.

Our main result is the following theorem that excludes a large class of attempts to prove Signed ElGamal CCA secure.

Theorem 1. *Suppose that DDH and IES hold in a cryptographic group G. Then there is no efficient key-passing black-box reduction from CCA security of Signed ElGamal to IND-CPA security of plain ElGamal in G.*

5 The Proof

We will construct a metareduction to IES from any CCA–to–IND-CPA reduction for Signed ElGamal. We introduce some variants of IES that will make the proof easier to present. These are not assumptions that we ask anyone to believe in directly: we show that they reduce to IES.

5.1 Verifiable IES

First, we deal with the issue that decrypted messages are not "verifiable". Proofs of knowledge are usually taken over NP relations (e.g. discrete logarithm). However, the statement that a ciphertext decrypts to a particular message is not immediately verifiable — it would require either the secret key or the encryption randomness to verify.

Our metareduction will have to check the decryptions produced by the reduction with which it interacts. We introduce a new assumption that we call verifiable IES or vIES to give the metareduction this ability; we also show that vIES reduces to IES. The new feature of vIES is that the adversary gets many attempts at guessing the message; formally we introduce a new oracle for the adversary to check messages.

Definition 3 (vIES). *The vIES problem in a cryptographic group* $(\mathbb{G}, q, g)$ *is to solve IES given the extra ability to check candidate solutions. Given* (g^x, g^r, g^a) *one may once submit a value* $c \in \mathbb{Z}_q$ *and learn* $s = a + cr \pmod q$ *in return; in addition, one may query an oracle* $\mathsf{check}(m)$ *many times which returns* 1 *if and only if* $m = g^{rx}$*. Ones win the game if one can find* g^{rx}*. A code-based presentation of the game is given in Figure 1.*

```
procedure initialise:
    $x \xleftarrow{\$} \mathbb{Z}_q$; $X \leftarrow g^x$;
    $r \xleftarrow{\$} \mathbb{Z}_q$; $R \leftarrow g^r$;
    $a \xleftarrow{\$} \mathbb{Z}_q$; $A \leftarrow g^a$;
    return $(X, R, A)$

oracle challenge($c$):
    return $a + cr \pmod q$
oracle check($m$):
    if $m = g^{rx}$ then return 1
    else return 0 endif
procedure finalise($m$):
    return check($m$)
```

Fig. 1: Verifiable IES. The checking oracle allows the adversary to test candidate solutions before submitting one. Challenge may only be called once.

The vIES assumption reduces to the IES assumption with a loss in soundness of a factor $k + 1$ where k is the number of checks made by the adversary. To see this, consider an efficient adversary with probability p of winning the vIES game and let k be a (polynomial) bound on the number of checks the adversary makes. Then with probability p, one of the following $k+1$ events occur: E_i for $1 \leq i \leq k$ is the event that the adversary makes at least i checking queries and the i-th check contains the correct message; E_0 is the event that the adversary never makes a checking query on the correct message but still calls the finalization oracle with the correct message.

Our reduction to IES guesses $i \xleftarrow{\$} \{0, 1, \ldots, k\}$ uniformly at random and simulates as follows: forward the initial data and the challenge query between the adversary and IES challenger, for $i > 0$ answer the first $i-1$ checking queries with 0 and pass the result of the i-th checking query to the IES finalization oracle directly, aborting the adversary at this point. For $i = 0$ answer 0 to all the adversary's checking queries and forward the adversary's output to the finalization oracle. If the adversary does not make i checking queries or in case 0 makes no output, abort.

If event E_i occurs then the reduction for case i will break IES. Since we assumed the adversary to succeed with probability p, at least one of the events will occur with probability $p/(k+1)$ as the $k+1$ events are a partition of the event that the adversary succeeds. Since the reduction chooses i uniformly, we conclude that it succeeds against IES with probability $p/(k+1)$.

5.2 One-more verifiable IES

For our metareduction we use a one-more variation of IES, for the same reason that Bernhard et al. 's proof that Fiat-Shamir-Schnorr is not adaptively secure requires the one-more discrete logarithm assumption. Unlike the cited theorem and assumption, the one-more IES assumption reduces to the basic one. We give the one-more assumption and reduction for verifiable IES; the same reduction holds for the non-verifiable variation.

The one-more assumption works as follows. The adversary may obtain and open a number of IES "instances"; her aim is to solve an unpoened instance. The initialization oracle produces a "public key" g^x shared between all instances. The instance oracle creates a new, fresh pair (g^r, g^a) together with an internal flag $f = 0$ to denote that this instance is fresh. The adversary may issue a challenge c once per instance, to which the challenger replies with $s = a + cr$ and sets the flag to $f = 1$ to denote that the challenge for this instance has been provided. In addition, the adversary may ask for an instance to be opened to which the challenger responds with (r, a) and sets $f = 2$. As in vIES, the adversary may also ask to check a value m against an instance, in which case the challenger reveals if $m = g^{rx}$. Checking does not affect the flag f. The adversary wins by providing the value $m = g^{rx}$ on an instance that has not been opened, i.e. $f \leq 1$. The one-more verifiable IES game is asymmetric in that the adversary must only solve a CDH instance to win but the game must provide a discrete logarithm r on request. It is this asymmetry that makes our metareduction work. Nonetheless, one-more verifiable IES reduces to plain IES. In the code-based presentation of the game in Figure 2, an index i is used to distinguish different instances.

Definition 4 (OMvIES). *The one-more verifiable IES game is given by the code in Figure 2.*

The reader may be asking why they should have any confidence that an assumption as complex as OMvIES should be hard. We note that vIES and OMvIES derive their justification solely from the fact that they reduce to IES: they are intermediate steps to make our main proof easier, not assumptions in their own right that we ask anyone to believe in. The justification for basic IES we gave when we introduced it, that a single Schnorr proof should not completely break the security of ElGamal encryption. The reason that the one-more version reduces to the simple one is that the instances are independent in the sense that the adversary cannot perform a challenge query that "touches" more than one instance.

Lemma 1. *There is a reduction from OMvIES to IES that loses a factor $O(k^2)$ in soundness where k is a bound on the number of queries made by the adversary.*

```
**procedure** initialise:
    $j \leftarrow 0$; $T \leftarrow [\,]$;
    $x \xleftarrow{\$} \mathbb{Z}_q$; $X \leftarrow g^x$;
    return $X$
**procedure** instance:
    $j \leftarrow j + 1$;
    $r \xleftarrow{\$} \mathbb{Z}_q$; $R \leftarrow g^x$;
    $a \xleftarrow{\$} \mathbb{Z}_q$; $A \leftarrow g^x$;
    $f \leftarrow 0$;
    $T[j] \leftarrow (r, a, f)$;
    return $(j, R, A)$
**procedure** finalise$(i, m)$:
    if $i > j$ then return 0 endif;
    $(r, a, f) \leftarrow T[i]$;
    if $f > 1$ then return 0 endif;
    return check$(i, m)$
**oracle** challenge$(i, c)$:
    if $i > j$ then return $\perp$ endif;
    $(r, a, f) \leftarrow T[i]$;
    if $f > 0$ then return $\perp$ endif;
    $T[i] \leftarrow (r, a, 1)$;
    return $a + cr \pmod q$
**oracle** check$(i, m)$:
    if $i > j$ then return 0 endif;
    $(r, a, f) \leftarrow T[i]$;
    if $m = g^{rx}$ then return 1
    else return 0 endif
**oracle** open$(i)$:
    if $i > j$ then return $\perp$ endif;
    $(r, a, f) \leftarrow T[i]$;
    $T[i] \leftarrow (r, a, 2)$;
    return $(r, a)$
```

Fig. 2: One-more verifiable IES. All IES instances share a common x but have their own r, a — which can be revealed using the open oracle.

Proof. It suffices to reduce OMvIES to vIES with a loss of $O(k)$. Given an upper bound k on the number of instances an adversary can create, pick $n \xleftarrow{\$} \{1, \ldots, k\}$ at random and use the vIES challenger for the n-th instance. Simulate all other instances by picking fresh (r, a). To open a simulated instance, simply reveal (r, a). To check a simulated instance against a candidate m, check if $m = X^r$. If the adversary tries to open the n-th instance, abort. If the adversary succeeds with probability p against OMvIES then she succeeds with probability at least p/k against the n-th instance, in which case she cannot have opened this instance. So the reduction wins the vIES game with at least p/k probability too. $\square$

5.3 A model adversary

Let $\mathcal{R}$ be a rewinding, black-box, key-passing reduction from CCA security of Signed ElGamal to IND-CPA security of plain ElGamal. That is, $\mathcal{R}$ may invoke multiple copies of a CCA adversary A which expects to receive a public key, can make one challenge and many decryption queries and will output a guess bit. $\mathcal{R}$ itself can interact with one IND-CPA challenger who provides a public key and a single challenge query, which returns a plain ElGamal ciphertext.

The aim of $\mathcal{R}$ is to guess its challenger's bit β. We construct an inefficient adversary A that breaks CCA of Signed ElGamal with advantage 1, that is it guesses correctly all the time. We will show that $\mathcal{R}$ cannot progress A to the point where it makes its guess, unless $\mathcal{R}$ invokes 2^n copies of A or breaks IES or

DDH. We also show how to construct an efficient simulation of (multiple copies of) A under these conditions.

Suppose w.l.o.g. that $q > 5$ and consider the inefficient adversary A_n in Figure 3 where $\Psi : \mathbb{Z}_q[X] \to \mathbb{Z}_q$ is a random function[5] (since efficiency is not an issue, random functions exist). RO is a random oracle call and dlog takes a discrete logarithm (which an inefficient adversary can also do). $\mathsf{Decrypt}$ and $\mathsf{Challenge}$ are calls to the CCA challenger.

```
// input: public key $Y$
$S \overset{\$}{\leftarrow} Z_q; T \leftarrow [\,]$

// PHASE 1 //
for $i = 1 \dots n$ do
    $r \leftarrow \Psi(1, S)$
    $a \leftarrow \Psi(2, S)$
    $m \leftarrow \Psi(3, S)$
    $M \leftarrow g^m$
    $(C, D) \leftarrow (g^r, MY^r)$
    $A \leftarrow g^a$
    $c \leftarrow \mathsf{RO}(Y, C, D, A)$
    $s \leftarrow a + cr \pmod{q}$
    $S \leftarrow (S, c)$
    $T[i] \leftarrow (M, C, D, A, s)$
endfor

// PHASE 2 //
for $i = n \dots 1$ step $(-1)$ do
    $(M, C, D, A, s) \leftarrow T[i]$
    $M' \leftarrow \mathsf{Decrypt}(C, D, A, s)$
    if $M \neq M'$ then abort endif
endfor

// PHASE 3 //
$m_0 \leftarrow \Psi(4, S)$
$m_1 \leftarrow \Psi(5, S)$
$(C, D, A, s) \leftarrow \mathsf{Challenge}(m_0, m_1)$
$r \leftarrow \mathsf{dlog}(C)$
$M \leftarrow D/Y^r$
if $M = g^{m_0}$ then return 0
else return 1 endif
```

Fig. 3: Adversary A_n against CCA of Signed ElGamal with advantage 1.

Adversary A_n runs in three phases. In phase 1, it builds up a chain of n Signed ElGamal ciphertexts in such a way that the randomness used in each ciphertext depends on the challenge returned from the random oracle in the previous one. Indeed, A_n only draws one random value to initialise S and uses the random function Ψ to update its state afterwards. One can think of S as the current state of a internal pseudorandom number generator.

In phase 2, our adversary asks decryption queries in reverse order, in the manner first proposed by Shoup and Gennaro [16] and used by Bernhard et al. [5]. Crucially, our adversary checks the correctness of each decrypt and aborts if the CCA game resp. reduction to which it is connected tries to cheat by returning a false decryption.

[5] By choosing the polynomial ring over $\mathbb{Z}_q$ as the domain, we mean that Ψ takes arbitrary-length finite sequences of integers modulo q as input.

By the time our adversary reaches phase 3, it is "satisfied" that whoever it is interacting with really can decrypt Signed ElGamal ciphertexts. It picks two random messages, asks a challenge query and takes a discrete logarithm to win the CCA game with overwhelming probability[6].

Our proof strategy will be to give an efficient simulation of phases 1 and 2 (which means dealing with Ψ) and to argue that no copy of A_n will ever reach phase 3 in less than exponential time, unless the reduction solves IES or DDH beforehand.

In the proof we will make three case distinctions. Recall that $\mathcal{R}$ is a reduction from CCA of Signed ElGamal to IND-CPA of plain ElGamal.

1. $\mathcal{R}$ answers the IND-CPA challenger's query without any copy of the adversary reaching Phase 3. In this case, we can simulate all copies of the adversary by lazily sampling the random function Ψ to obtain an IND-CPA adversary that wins its game with the same probability as $\mathcal{R}$ given access to a CCA adversary that always guesses correctly.
2. $\mathcal{R}$ answers a decryption query on a ciphertext without using special soundness. We build a metareduction to IES.
3. Neither of the above cases occur. In this case one copy of the adversary we are simulating proceeds to the point where it would have to use its discrete logarithm capability hence it must have got answers to all n decryption queries. In this case we show that the reduction must have launched $\Omega(2^n)$ copies of the adversary.

5.4 Case 1: the reduction solves DDH by itself

If the reduction answers its IND-CPA challenge without getting any copy of the adversary to run to phase 3 then the reduction must be breaking indistinguishability "by itself". In this case we can just simulate the adversary efficiently for as long as needed.

Lemma 2. *Let E_1 be the event that the reduction $\mathcal{R}$ returns a guess to its challenger without any copy of the adversary reaching Phase 3. There is a metareduction M_1 that breaks DDH in $\mathbb{G}$ with advantage $\alpha_M = \Pr[E_1]\alpha_{E_1}/2$ where α_{E_1} is the advantage of $\mathcal{R}$ (with access to our adversary) given that E_1 has occurred.*

Proof. Consider an efficient metareduction M_1 which simulates all the copies of our adversary in phases 1 and 2 and the random function by lazy sampling, once for all copies of the adversary. If an adversary copy reaches phase 3 or $\mathcal{R}$ aborts, M_1 outputs a random guess. Writing $\sigma_{E_1} := \Pr[\mathcal{R}$ guesses correctly $\mid E_1]$ and $\alpha_{E_1} := (2\sigma_{E_1} - 1)$ we compute the advantage of M_1 as $\Pr[E_1] \cdot \alpha_{E_1}$. The advantage against the encryption scheme gives an adversary against DDH with advantage $\Pr[E_1]\alpha_{E_1}/2$. $\square$

[6] The probability is not 1 because m_0 and m_1 could collide.

5.5 Case 2: The reduction breaks IES

If the reduction $\mathcal{R}$ does run a copy of the adversary to phase 3, we can hope that it solves IES for us along the way. We define a metareduction M_2 as follows. Our metareduction M_2 simulates both the adversary and challenger interfaces towards the reduction $\mathcal{R}$ and interacts with an OMvIES challenger. On the challenger interface M passes the challenger's public key. By assumption, $\mathcal{R}$ is key-passing so although the simulated adversaries formally receive a public key from $\mathcal{R}$ we could equally well have the metareduction provide them with this key directly. If the reduction asks an IND-CPA challenge query, we just simulate this challenge query (picking a random bit b); since we have already dealt with case 1 we can ignore the reduction returning a guess to the challenger for now.

In detail, our metareduction operates as follows. Initially, it obtains a value X from its OMvIES challenger and hands control to the reduction $\mathcal{R}$. When $\mathcal{R}$ asks to invoke a new copy of the adversary A, metareduction M simulates a copy of A using public key X (since $\mathcal{R}$ is key-passing) using the algorithms in Figure 4. The oracles check, draw and chal are shared between all copies of the simulated adversary. $\mathsf{R.alg}$ means we call back to $\mathcal{R}$, simulating the adversary calling its challenger's oracle named alg whereas $\mathsf{I.alg}$ means call the oracle named alg on the OMvIES challenger.

If the reduction makes a challenge query (to its IND-CPA challenger) the metareduction draws a random bit b and simulates the challenge ciphertext; if the reduction $\mathcal{R}$ makes a guess at b then the metareduction aborts (this is case 1 which we have dealt with above). If $\mathcal{R}$ manages to get a copy of the simulated adversary to phase 3, the metareduction M_2 aborts too — this is case 3 which we will deal with later.

The check, draw and chal oracles help the metareduction M_2 simulate multiple copies of A_n using only one OMvIES challenger. The draw oracle ensures that multiple copies of the adversary who receive identical (random oracle) replies from $\mathcal{R}$ also produce identical ciphertexts. In a table U the metareduction keeps track of whether a particular adversary state S has been encountered before; if so we can simply replay the same ciphertexts.

The chal oracle is responsible for completing the proofs in Signed ElGamal ciphertexts. The table V maps each OMvIES instance (R, A) to the following parameters:

- The integer j is the index required to tell the challenger to operate on this particular instance.
- The potential ϕ is the equivalent of the OMDL potential in Bernhard et al. 's proof [5] and matches the potential f stored internally by the OMvIES challenger.
  - The first time a particular instance (R, A) is used (case $\phi = 0$), chal uses the OMvIES challenge oracle to complete the proof.
  - In case $\phi = 1$, if the current challenge has been used before then the reduction is replaying one adversary copy's responses to a second copy. In this case the metareduction replays the response s that it computed

earlier. If c is fresh on the other hand, then the reduction has "forked" two copies of the adversary on the random oracle call in this proof and is about to recover the discrete logarithm r by applying special soundness. In this case our metareduction M_2 opens the instance.
  - In case $\phi = 2$ the instance has already been opened, so M_2 knows the values a, r necessary to make the proof itself.
- The values c', s' in a V-entry store the challenge and response from a previous chal query. These values are used in case $\phi = 1$ and the reduction replays the same c', in which case the metareduction replies with the same s'.
- The values r', a' store the discrete logarithms of R, A when $\phi = 2$. In this case the reduction has forked the adversary on the instance (R, A), forcing the metareduction to open the instance.

The check oracle is responsible for checking both that the reduction does not cheat and whether the reduction has solved OMvIES for us. When the reduction returns a decryption M to a copy of the adversary, the simulated adversary strips out the message to recover what would be the CDH solution g^{rx} for the instance in question. The metareduction M_2 then checks this with the OMvIES challenger. Should the decryption turn out to be false, the copy of the adversary in question aborts. If the decryption is correct and the potential is not yet at 2 then the reduction has given us some information that we do not know already (the instance in question is unopened) and we solve OMvIES.

The above arguments show that whenever $\mathcal{R}$ decrypts an unopened challenge instance, the metareduction M_2 breaks OMvIES. It remains to show that $\mathcal{R}$ cannot distinguish M from multiple, independent copies of A_n running on the same random string. Recall that in Figure 3 we have the following invariants.

1. Two copies of A_n that receive identical messages from $\mathcal{R}$ also produce identical messages/calls back to $\mathcal{R}$.
   This is because all copies execute on the same random string and do not communicate with anyone except $\mathcal{R}$.
2. $\mathcal{R}$ can influence copies of A_n in exactly three places: answering random oracle queries in phase 1, decryption queries in phase 2 and the challenge query in phase 3.
3. The moment that two copies of A_n get different answers to a random oracle query, the two copies become independent of each other.
   If at some point two copies U, V get different answers $c_U \neq c_V$ to the same random oracle query then their states S_U, S_V will become distinct from then on and never coincide again (since we only ever append to state vectors). Since the randomness used to construct Signed ElGamal ciphertexts is drawn using a random function Ψ from the current state S, the ciphertexts in two copies with different states are independent.
4. The distribution of each individual ElGamal ciphertext and Schnorr commitment produced by A_n is uniform, that is (C, D, A) is a uniformly random element of $\mathbb{G}^3$.
   This follows from r, a, m being drawn by a random function on distinct inputs.

5. In phase 2, a copy of A_n will proceed past a decryption query (and not abort) if and only if the decryption is correct.

These invariants will let us show that the values received by $\mathcal{R}$ when interacting with M_2 or multiple copies of A_n are identically distributed. We use induction over the sequence of all calls made to $\mathcal{R}$ in a particular execution. Before the first call, the distributions of all values sent to $\mathcal{R}$ are certainly equal.

– For a RO call, there are two cases. If this call is made by a copy of the adversary that has received the exact same sequence of inputs and outputs as some other copy has received previously, then it will return the same values (X, C, D, A) as the previous copy.
  This holds for A_n as the state S of the copy that sent the current call will match the state S' of the previous copy at the time it sent the equivalent call, so the values C, D, A will be equal (and X is constant in any case).
  In M_2, the oracle draw ensures that the same state S leads to the same values (R, A, d) being returned.
– For a fresh RO call (that does not match the case above), the value X is constant and the values R, D, A are uniformly random and independent of each other and all values sent to the reduction $\mathcal{R}$ so far.
  In A_n this holds because S is fresh and the values in question are therefore obtained by a random function on distinct, fresh inputs (since m is uniform, so is M and because M is not used elsewhere, so is MY^r). In M_2 a fresh S causes (R, A) to be sampled from the OMvIES challenger so they are uniform and independent of previous values as expected; D is also a fresh, uniform group element.
– The value s in a decryption query is completely determined by the matching C, D, A and c — all of which $\mathcal{R}$ has seen before in the matching random oracle query, or in the case of c the reduction $\mathcal{R}$ has chosen the value itself. This holds in both A_n and M_2.

It follows that up until some adversary copy reaches phase 3, $\mathcal{R}$ cannot tell M_2 from A_n and must therefore have a negligibly close IND-CPA advantages in both experiments.

5.6 Case 3: the reduction takes exponential time

This case is essentially the same argument as that of Bernhard et al. [5]. If the reduction $\mathcal{R}$ when interacting with M_2 ever gets a copy of the simulated adversary to phase 3 (in which case M_2 aborts) then it must have launched at least 2^n copies of the adversary.

Lemma 3. *Consider an execution of M_2 with any reduction $\mathcal{R}$ that results in one copy of the simulated adversary advancing to phase 3. Then $\mathcal{R}$ must have launched 2^n copies of the adversary.*

If a copy of the adversary simulated by M_2 advances to phase 3, we know that neither has $\mathcal{R}$ returned a guess at β (this would have halted the entire execution as in Case 1) nor has the check oracle aborted because OMvIES has been solved (Case 2). In particular, $\mathcal{R}$ has never answered a decryption query on a ciphertext linked to an OMvIES instance at potential $\phi \leq 1$.

We build a complete binary tree of depth n representing instances of the adversary that $\mathcal{R}$ must have intereacted with. Nodes will have the form (i, k) where i is an identifier for some copy of the adversary (for example, one can number the copies in the order that they begin phase 1) and $k \leq n$. Our nodes will have the following invariants.

1. Any copy of the adversary referenced in the tree has advanced to at least phase 2 and has obtained all its n challenges. If a node (i, k) is present then copy i has also obtained answers to at least its first k decryption queries,
2. The root of the tree is of the form (i, n). A child of (i, k) is of the form $(j, k-1)$ and a descendant of (i, k) is of the form (j, l) with $0 \leq l < k$.
3. If (j, l) is a descendant of (i, k) then (1) the copy j has got the same first $n - k$ challenges to copy i. (The two could also be identical.) However, (2) the copies $(j, k-1)$ and $(j', k-1)$ represented by the two children of (i, k) differ in challenge $n - k + 1$.

If we can construct such a complete binary tree rooted at (i, n) where i is the copy of the adversary that reached phase 3 then we claim that all 2^n leaves of this tree represent distinct copies of the adversary, proving our exponential lower bound. Suppose for the sake of contradiction that two distinct leaves $L = (j, 0)$ and $M = (j', 0)$ refer to the same copy of the adversary, i.e. $j = j'$. Then consider the unique path from the one leaf to the other in the tree, and the highest (i.e. closest to the root) node $R = (i, k)$ on this path. R will have two children A and B, since R is not itself a leaf by construction. W.l.o.g. L is a descendant or equal to A and M is a descendant or equal to B. The contradiction is that by invariant 3, A and B must differ in their $n - k + 1$st challenge (part 1 of the invariant) whereas all descendants of A, including L, must share their $n-k+1$st challenge with A (by applying part 2 of the invariant to A). Similarly, M must share challenge $n - k + 1$ with B and therefore L, M must differ in challenge $n - k + 1$. It follows that $j \neq j'$ and that there must be 2^n distinct copies of the adversary referenced in the leaves, hence $\mathcal{R}$ must have launched this many copies.

To construct the tree we pick the copy i that reached phase 3 and use (i, n) as the root; this trivially meets all invariants. We repeatedly give each node (j, l) with $l > 0$ two children as follows. The first child of (j, l) is simply $(j, l - 1)$. Invariant 1 carries over as the second component of the node decreases, invariants 2 and 3(1) are trivially satisfied.

The core of the tree construction is in the choice of the second child for each node. For the second child of (j, l) with $l > 0$ we observe that since copy j has got an answer to its l-th decryption query yet M_2 has not solved OMvIES with this answer, the corresponding IES instance must be at $\phi = 2$. Therefore some other copy j' of the adversary must have triggered the opening of this instance, before copy j got its l-th decryption query answered. This other copy j' must therefore have shared challenges 1 up to $n-l$ with j and been "forked" on challenge $n-l+1$ to open the IES instance in question. And this forking can only have happened after j' had its own $l-1$st decryption query answered, since it must have been the lth decryption query of j' that triggered the opening. It follows that we can pick $(j', l-1)$ as our second child of (j, l) to satisfy all the invariants.

Taken together, our three cases show that a key-passing black-box reduction $\mathcal{R}$ from IND-CPA security of Signed ElGamal to IND-CPA security of plain ElGamal must either solve DDH, or IES, or run in exponential time. This proves our Theorem 1. □

6 Conclusion

Our impossibility result, just as previous metareduction results, does not unconditionally rule out reductions; the result uses the fact that the reduction uses black-box access to the adversary in a fundamental way. Other requirements of our result concern the necessity of the IES assumption to hold, and the fact that the reduction must be key-passing. Refuting the IES assumption seems to be hard to us, because one would expect that the interactive Schnorr proof does not facilitate the computation of the CDH problem. As for the key-passing property, this seems to be a common way to build a reduction to the CPA security of ElGamal, and has been used in the proofs of [18,13] with some additional knowledge assumption. Overall, we thus take our result as a good indication that Signed ElGamal cannot be proven CCA-secure in the random oracle model (only).

```
// COPY OF $A_n$ //

// PHASE 1 //
$S \leftarrow 0$
for $i = 1 \dots n$ do
    $(R, A, d) \leftarrow \mathsf{draw}(S)$
    $c \leftarrow \mathsf{R.RO}(X, R, g^d, A)$
    $s \leftarrow \mathsf{chal}(R, A, c)$
    $S \leftarrow (S, c)$
    $T[i] \leftarrow (R, A, d, c, s)$
endfor

// PHASE 2 //
for $i = n \dots 1$ step $(-1)$ do
    $(R, A, d, c, s) \leftarrow T[i]$
    $M \leftarrow \mathsf{R.decrypt}(R, g^d, A, s)$
    $Z \leftarrow g^d / M$
    if not $\mathsf{check}(R, A, Z)$ then
        abort this copy of $A_n$
    endif
endfor

oracle check$(R, A, Z)$:
    $(j, \phi, c', s', r', a') \leftarrow V[R, A]$
    $\beta \leftarrow \mathsf{I.check}(j, Z)$
    if $\beta = 1$ and $\phi < 2$ then
        abort and return $(j, Z)$
        to OMvIES challenger
    else
        return $\beta$
    endif

oracle draw$(S)$:
    if $U[S]$ is defined then
        $(R, A, d) \leftarrow U[S]$
        return $(R, A, d)$
    else
        $(j, R, A) \leftarrow \mathsf{I.Instance}()$
        $d \xleftarrow{\$} \mathbb{Z}_q$
        $U[S] \leftarrow (R, A, d)$
        $V[R, A] \leftarrow (j, 0, 0, 0, 0, 0)$
        return $(R, A, d)$
    endif

oracle chal$(R, A, c)$:
    $(j, \phi, c', s', r', a') \leftarrow V[R, A]$
    if $\phi = 0$ then
        $s \leftarrow \mathsf{I.Challenge}(j, c)$
        $V[R, A] \leftarrow (j, 1, c, s, 0, 0)$
        return $s$
    elseif $\phi = 1$ then
        if $c = c'$ then // replay
            return $s'$
        else // fork
            $(r, a) \leftarrow \mathsf{I.open}(j)$
            $V[R, A] \leftarrow (j, 2, c', s', r, a)$
            return $a + cr \pmod q$
        endif
    else // $\phi = 2$
        return $a' + cr' \pmod q$
    endif
```

Fig. 4: How to simulate the adversary A_n for an OMvIES reduction.

References

1. M. Abdalla, F. Benhamouda and P. MacKenzie. Security of the J-PAKE Password-Authenticated Key Exchange Protocol. To appear in S&P 2015.
2. B. Adida. Helios: Web-based open-audit voting. In: 17th USENIX security symposium, Pages 335-348, 2008. Helios website: `http://heliosvoting.org` paper: `http://www.usenix.org/events/sec08/tech/full_papers/adida/adida.pdf`
3. Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. ACM Conference on Computer and Communications Security, pages 62-73, ACM, 1993.
4. Relations Among Notions of Security for Public-Key Encryption Schemes. M. Bellare, A. Desai, D. Pointcheval and P. Rogaway. In: Advances in Cryptology (CRYPTO '98), LNCS 1462, pages 26–45, 1998.
5. David Bernhard, Marc Fischlin and Bogdan Warinschi. Adaptive Proofs of Knowledge in the Random Oracle Model. PKC '15, LNCS 9020, Springer, pages 629âĂŞ649, 2015.
6. D. Bernhard, O. Pereira and B. Warinschi. How not to Prove Yourself: Pitfalls of Fiat-Shamir and Applications to Helios. In: Advances in Cryptology — Asiacrypt '12, LNCS 7658, pages 626–643, 2012.
7. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Proceedings of the 18th Annual International Cryptology Conference on Advances in Cryptology (CRYPTO '98), pages 13–25, 2008.
8. Danny Dolev, Cynthia Dwork, and Moni Naor, Non-Malleable Cryptography, SIAM Journal on Computing, pages = 542–552, 2000.
9. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In: Proceedings on advances in cryptology (CRYPTO '86), pages 186–194, 1986.
10. M. Fischlin. Communication-Efficient Non-Interactive Proofs of Knowledge with Online Extractors. In: Proceedings of the 25th annual international cryptology conference on advances in cryptology (CRYPTO '05), pages 152–168, 2005.
11. David Pointcheval and Jacques Stern. Security Arguments for Digital Signatures and Blind Signatures. J. Cryptolog, volume 13, number 3, Springer, pages 361-396, 2000.
12. C.P. Schnorr. Efficient signature generation for smart cards. In: Journal of cryptology, Volume 4, Springer, pages 161–174, 1991.
13. C.P. Schnorr and M. Jakobsson. Security of Signed ElGamal Encryption. In: Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT '00), Springer, pages 73–89, 2000.
14. Y. Seurin and J. Treger. A Robust and Plaintext-Aware Variant of Signed ElGamal Encryption. CT-RSA, Lecture Notes in Computer Science, volume 7779, Springer, pages 68-83, 2013.
15. V. Shoup. A Proposal for an ISO Standard for Public Key Encryption. Version 2.1. Available at `www.shoup.net`, 2001.
16. V. Shoup and R. Gennaro. Securing Threshold Cryptosystems Agains Chosen-Ciphertext Attack. In: Advances in Cryptology (Eurocrypt '98), LNCS 1403, pages 1–16, 1998.
17. Victor Shoup and Rosario Gennaro. Securing Threshold Cryptosystems against Chosen Ciphertext Attack. J. Cryptology, volume 15, number 2, Springer, pages 75-96, 2002.

18. Y. Tsiounis and M. Yung. On the security of ElGamal-based encryption. In: International Workshop on Practice and Theory in Public Key Cryptography (PKC '98), pages 117–134, 1998.
19. D. Wikström. Simplified Submission of Inputs to Protocols. In: Security and Cryptography for Networks, 6th International Conference, SCN 2008, pages 293–308, 2008.