On CCA1-Security of Elgamal And Damgård's Elgamal

Helger Lipmaa

Cybernetica AS, Estonia, `http://research.cyber.ee/~lipmaa`

Abstract. We establish the complete complexity landscape surrounding CCA1-security of Elgamal and Damgård's Elgamal (DEG). Denote by $X^{Y[i]}$ the assumption that the adversary, given a non-adaptive oracle access to the Y oracle with i free variables cannot break the assumption X. We show that the CCA1-security of Elgamal is equivalent to the $\mathrm{DDH}^{\mathrm{CDH}[1]}$ assumption. We then give a simple alternative to Gjøsteen's proof that DEG cryptosystem is CCA1-secure under the $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption. We also provide several separations. We show that DDH cannot be reduced to $\mathrm{DDH}^{\mathrm{CDH}[1]}$ in the generic group model. We give an irreduction showing that DDH cannot be reduced to $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ (unless DDH is easy), $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ cannot be reduced to $\mathrm{DDH}^{\mathrm{DDH}[2]}$ (unless $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ is easy) and $\mathrm{DDH}^{\mathrm{DDH}[2]}$ cannot be reduced to the $\mathrm{DDH}^{\mathrm{CDH}[1]}$(unless $\mathrm{DDH}^{\mathrm{DDH}[2]}$ is easy). All those irreductions are optimal in the sense that they show that if assumption X can be reduced to Y in polynomial time then X has to be solvable in polynomial time itself and thus *both* assumptions are broken.
Keywords. CCA1-security, DEG cryptosystem, DDH, Elgamal cryptosystem, irreduction.

1 Introduction

While the Elgamal cryptosystem [Elg85] is one of the best-known public-key cryptosystems, results on its security have been slow to come. Only in 1998, it was proven that Elgamal is CPA-secure [TY98], that is, secure against chosen plaintext attacks. On the other hand, it is clearly not CCA2-secure, that is, secure against adaptive chosen ciphertext attacks, because it is homomorphic. However, not much is known about its CCA1-security, that is, security against non-adaptive chosen ciphertext attacks.

Denote by $X^{Y[i]}$ the assumption that no adversary, given a non-adaptive oracle access to the Y oracle with i free variables can break the assumption X (w.r.t. some fixed ordering of the variables). We show that Elgamal is CCA1-secure if and only if the $\mathrm{DDH}^{\mathrm{CDH}[1]}$ assumption is true. In some sense, this result just states that Elgamal is CCA1-secure if and only if it is CCA1-secure, but it serves as an introduction to the second main result. Moreover, this is the weakest assumption up to now under which Elgamal has been proven CCA1-secure.

In 1991, Damgård proposed what we call the DEG (Damgård's Elgamal) cryptosystem [Dam91] and proved it to be CCA1-secure under a knowledge-of-the-exponent assumption. Only recently, Gjøsteen proved [Gjø06] that DEG is CCA1-secure under a more standard $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption. Gjøsteen's proof consisted of a relatively long chain of games.

OUR CONTRIBUTIONS. In the current paper, we establish the complete complexity landscape of CCA1-security of Elgamal and DEG. We establish precise security assumptions under which these cryptosystems are CCA1-secure. Moreover, we show that the CCA1-security of DEG does not follow from DDH (unless DDH is the concrete group is easy) and that the CCA1-security of Elgamal does not follow from $\mathrm{DDH}^{\mathrm{DDH}[2]}$ and thus from the CCA1-security of DEG (unless $\mathrm{DDH}^{\mathrm{DDH}[2]}$ is not true). Moreover, we give a simpler proof of Gjøsteen's result and show that $\mathrm{DDH}^{\mathrm{DDH}[2]}$ does not follow from the CCA1-security of DEG either (unless DEG is insecure), and is thus unnecessarily strong. The rest of the introduction gives a more technical exposure of our results.

We first give a simple proof that DEG is CCA1-secure if and only if the $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ assumption holds, where both CDEG and DDEG are new but standard-looking assumptions. We also show that one can derive the $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ assumption under the more traditional $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption. The proof is a simple hybrid argument following the general guideline "if $X \Rightarrow X'$ and $Y' \Rightarrow X'$ then $X^Y \Rightarrow (X')^{Y'}$", though it also uses a recent trapdoor trick from [CKS08]. (Here and in what follows, $X \Rightarrow Y$ means that the assumption Y can be reduced to the assumption X.) Both this and analogous direct proof that DEG is CCA1-secure under the $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption consist of two game hops, and are thus considerably simpler than the proof given in [Gjø06]. Our proof technique may be a contribution by itself.

We then concentrate on showing that the used assumptions are all (potentially) different. Following the approach of [Che06], we show that in the generic group model, $\mathrm{DDH}^{\mathrm{CDH}[1]} \not\Rightarrow \mathrm{DDH}$. That is, we show that there is no reduction from $\mathrm{DDH}^{\mathrm{CDH}[1]}$ to DDH. The corresponding irreducibility notion in the generic group model seems to be novel. This result also means that CCA1-security of Elgamal cannot directly be proven from the DDH assumption alone by using low-degree polynomial reductions. It is not clear whether this is true for the DEG.

In addition, we give an irreduction [Bro07,BMV08] showing that DDH cannot be reduced to $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ (unless DDH is easy), $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ cannot be reduced to $\mathrm{DDH}^{\mathrm{DDH}[2]}$ (unless $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ is easy) and $\mathrm{DDH}^{\mathrm{DDH}[2]}$ cannot be reduced to the $\mathrm{DDH}^{\mathrm{CDH}[1]}$(unless $\mathrm{DDH}^{\mathrm{DDH}[2]}$ is easy). All those irreductions are optimal in the sense that they show that if assumption X can be reduced to Y in polynomial time then X has to be solvable in polynomial time itself and thus *both* assumptions are broken.

That is, we prove that:

$$\boxed{\text{Elgamal-CCA1} \underset{\Rightarrow}{\overset{\Leftarrow}{\ }} \mathrm{DDH}^{\mathrm{CDH}[1]} \underset{\not\Rightarrow_{\mathrm{DDH}^{\mathrm{DDH}[2]}}}{\overset{\Leftarrow}{\ }} \mathrm{DDH}^{\mathrm{DDH}[2]} \underset{\not\Rightarrow_{\mathrm{DDEG}^{\mathrm{CDEG}[2]}}}{\overset{\Leftarrow}{\ }} \mathrm{DDEG}^{\mathrm{CDEG}[2]} \underset{\Rightarrow}{\overset{\Leftarrow}{\ }} \text{DEG-CCA1} \underset{\not\Rightarrow_{\mathrm{DDH}}}{\overset{\Leftarrow}{\ }} \mathrm{DDH}}$$

Therefore, we give a complete map of the security reductions and irreductions between these security assumptions.

ROAD-MAP. In Sect. 2, we lay out the preliminaries. In Sect. 3, we study the CCA1-security of DEG. In Sect. 4, we study the CCA1-security of Elgamal. Finally, in Sect. 5, we provide irreductions between DDH, the CCA1-securities of DEG and Elgamal, and $\mathrm{DDH}^{\mathrm{DDH}[2]}$.

2 Preliminaries

2.1 Reductions And Irreductions

We say that security assumption Y can be reduced to assumption X, $X \Rightarrow Y$, if there exists a *reduction* $\mathcal{R}$, such that: for every adversary $\mathcal{A}$ that breaks assumption X, $\mathcal{R}$ can break assumption Y by using $\mathcal{A}$ as an oracle. Following [Bro07], we call an algorithm $\mathcal{I}$ an *irreduction* $X \not\Rightarrow_Y Z$, if it can, given as an oracle an arbitrary reduction algorithm $X \Rightarrow Z$, solve problem Y. If $Y = Z$ then we say that $\mathcal{I}$ is an *optimal irreduction* algorithm and write $X \not\Rightarrow_! Z$.

2.2 Assumptions

Let the value of the predicate $[a \overset{?}{=} b]$ be 1, if $a = b$, and 0 otherwise. Denote

$$\mathrm{cdh}(g, g^x, g^y) := g^{xy} \quad ,$$
$$\mathrm{ddh}(g, g^x, g^y, g^z) := [g^z \overset{?}{=} \mathrm{cdh}(g, g^x, g^y)] \quad .$$

Fix a group $\mathbb{G} = \langle g \rangle$ of order q. The CDH *game* is defined as follows:

Setup phase. Challenger sets $\mathrm{sk} \leftarrow \mathbb{Z}_q$, $\mathrm{pk} \leftarrow g^{\mathrm{sk}}$. He sends pk to the adversary $\mathcal{A}$.
Challenge phase. Challenger sets $b_{\mathcal{A}} \leftarrow \{0, 1\}$, $y^* \leftarrow \mathbb{Z}_q$, $h^* \leftarrow g^{y^*}$. He sends h^* to $\mathcal{A}$.
Guess phase. $\mathcal{A}$ returns a group element $h^*_{\mathcal{A}} \in \mathbb{G}$. $\mathcal{A}$ wins if $h^*_{\mathcal{A}} = \mathrm{cdh}(g, \mathrm{pk}, h^*)$, that is, if $h^*_{\mathcal{A}} = \mathrm{pk}^{y^*}$.

Group $\mathbb{G}$ is a (τ, ε)-CDH *group* if for any adversary $\mathcal{A}$ working in time τ, $\Pr[\mathcal{A}$ wins in the CDH game$] \leq \frac{1}{q} + \varepsilon$.

Fix a group $\mathbb{G} = \langle g \rangle$ of order q. The DDH *game* is defined as follows:

Setup phase. Challenger sets $\mathrm{sk} \leftarrow \mathbb{Z}_q$, $\mathrm{pk} \leftarrow g^{\mathrm{sk}}$. He sends pk to the adversary $\mathcal{A}$.
Challenge phase. Challenger sets $b_{\mathcal{A}} \leftarrow \{0, 1\}$, $y^* \leftarrow \mathbb{Z}_q$, $z^* \leftarrow \mathbb{Z}_q$, $h^*_1 \leftarrow g^{y^*}$. He sets $h^*_2 \leftarrow g^{z^*}$ if $b_{\mathcal{A}} = 0$ and $h^*_2 = \mathrm{cdh}(g, \mathrm{pk}, h^*_1) = \mathrm{pk}^{y^*}$ if $b_{\mathcal{A}} = 1$. Challenger sends (h^*_1, h^*_2) to $\mathcal{A}$.
Guess phase. $\mathcal{A}$ returns a bit $b_{\mathcal{A}} \in \{0, 1\}$. $\mathcal{A}$ wins if $b'_{\mathcal{A}} = b_{\mathcal{A}}$, that is, if $b'_{\mathcal{A}} = \mathrm{ddh}(g, \mathrm{pk}, h^*_1, h^*_2)$.

Group $\mathbb{G}$ is a (τ, ε)-DDH *group* if for any adversary $\mathcal{A}$ working in time τ, $\Pr[\mathcal{A}$ wins in the DDH game$] \leq \frac{1}{2} + \varepsilon$.

Based on arbitrary assumptions $X = X(x_1, \dots, x_m)$ and $Y = Y(y_1, \dots, y_n)$ we define a new assumption $X^{Y[i]}$. In the $X^{Y[i]}$ game, adversary has oracle access to an oracle solving assumption Y with parameters y_j, and she has to break a random instance of the X assumption with parameters x_i. In addition, the queries are restricted so that only the i last variables $(y_{n-i+1}, \dots, y_n)$ in any query are chosen by the adversary, while the first variables $(y_1, \dots, y_{n-i})$ must coincide with the first variables $(x_1, \dots, x_{n-i})$ of the instance of X assumption she is trying to solve. We denote $X^{Y[n]}$ just by X^Y. For example, $\mathrm{DDH}^{\mathrm{DDH}} = \mathrm{DDH}^{\mathrm{DDH}[4]}$.

A group is (τ, ε)-$X^{Y[i]}$ *group* if for any adversary $\mathcal{A}$ working in time τ, $\Pr[\mathcal{A}$ wins in the $X^{Y[i]}$ game$] \leq \frac{1}{\delta} + \varepsilon$, where δ is an assumption-dependent constant (usually $\frac{1}{2}$ or $\frac{1}{q}$). Note that the order of the arguments is important, and thus we have to always explicitly define X and Y. Many versions of the $X^{Y[i]}$ game for different values of X, Y and i, have been used before. For the sake of clarity, we now give a precise definition of the $\mathrm{DDH}^{\mathrm{DDH}[2]}$ game, and we state its relation to some of the existing assumptions.

Fix a group $\mathbb{G} = \langle g \rangle$ of order q. The $\mathrm{DDH}^{\mathrm{DDH}[2]}$ *game* is defined as follows:

Setup phase. Challenger sets $\mathrm{sk} \leftarrow \mathbb{Z}_q$, $\mathrm{pk} \leftarrow g^{\mathrm{sk}}$. He sends pk to adversary $\mathcal{A}$.
Query phase. $\mathcal{A}$ has an access to oracle $\mathrm{ddh}(g, \mathrm{pk}, \cdot, \cdot)$.
Challenge phase. Challenger sets $b_{\mathcal{A}} \leftarrow \{0,1\}$, $y^*, z^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$. He sets $h_2^* \leftarrow g^{z^*}$ if $b_{\mathcal{A}} = 0$ and $h_2^* \leftarrow \mathrm{pk}^{y^*} = \mathrm{cdh}(g, \mathrm{pk}, h_1^*)$ if $b_{\mathcal{A}} = 1$. Challenger sends (h_1^*, h_2^*) to $\mathcal{A}$.
Guess phase. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}} \in \{0,1\}$. $\mathcal{A}$ wins if $b'_{\mathcal{A}} = b_{\mathcal{A}}$, that is, if $b'_{\mathcal{A}} = \mathrm{ddh}(g, \mathrm{pk}, h_1^*, h_2^*)$.

Group $\mathbb{G}$ is a (τ, ε)-$\mathrm{DDH}^{\mathrm{DDH}[2]}$ *group* if for any adversary $\mathcal{A}$ working in time τ, $\Pr[\mathcal{A}$ wins in the $\mathrm{DDH}^{\mathrm{DDH}[2]}$ game$] \leq \frac{1}{2} + \varepsilon$.

Clearly, $X^{Y[i]} \Rightarrow (X')^{Y'[i']}$ when $X \Rightarrow X'$, $Y' \Rightarrow Y$ and $i \geq i'$. This can be proven by using a hybrid argument, showing say that $X^{Y[i]} \Rightarrow X^{Y'[i]}$, that $X^{Y'[i]} \Rightarrow (X')^{Y'[i]}$, etc. We use such a hybrid argument later.

The gap DH assumption of [OP01] is equal to $\mathrm{CDH}^{\mathrm{DDH}[3]}$. The strong DH assumption of [ABR01] is slightly different, giving first access to h_1^* and the oracle $\mathrm{ddh}(g, \cdot, \cdot, \mathrm{sk})$, and then asking to compute $\mathrm{cdh}(g, \mathrm{pk}, h_1^*)$. $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption has been used before say in [Gjø06]. Some other papers deal with the so called *one-more DDH* assumption, where $\mathcal{A}$ has to answer correctly to $t+1$ DDH challenges after making only t DDH queries. See, for example, [Bro07].

2.3 Cryptosystems

A public-key cryptosystem Π is a triple of efficient algorithms (G, E, D), where $G(1^k)$ outputs a key pair $(\mathrm{sk}, \mathrm{pk})$, $E_{\mathrm{pk}}(m; r)$ returns a ciphertext and $D_{\mathrm{sk}}(c)$ returns a plaintext, so that $D_{\mathrm{sk}}(E_{\mathrm{pk}}(m; r)) = m$ for any $(\mathrm{sk}, \mathrm{pk}) \in G(1^k)$. Here, k is a security parameter that we will just handle as a constant.

Fix a cyclic group $\mathbb{G} = \langle g \rangle$ of order q. The *Elgamal cryptosystem* [Elg85] in group $\mathbb{G}$ is defined as follows:

Key generation $G(1^k)$. Select a random $\mathrm{sk} \leftarrow \mathbb{Z}_q$, set $\mathrm{pk} \leftarrow g^{\mathrm{sk}}$. Publish pk.
Encryption $E_{\mathrm{pk}}(m; \cdot)$. Return $\perp$ if $m \notin \mathbb{G}$. Otherwise, select a random $r \leftarrow \mathbb{Z}_q$, set $E_{\mathrm{pk}}(m; r) \leftarrow (g^r, m \cdot \mathrm{pk}^r)$.
Decryption $D_{\mathrm{sk}}(c)$. Parse $c = (c_1, c_2)$, return $\perp$ if $c_i \notin \mathbb{G}$ for some i. Otherwise, return $D_{\mathrm{sk}}(c) \leftarrow c_2 / c_1^{\mathrm{sk}}$.

Fix a group $\mathbb{G} = \langle g \rangle$ of order q. The *Damgård's Elgamal (DEG) cryptosystem [Dam91]* in group $\mathbb{G}$ is defined as follows:

Key generation $G(1^k)$. Select random $\mathrm{sk}_1, \mathrm{sk}_2 \leftarrow \mathbb{Z}_q$, set $\mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}, \mathrm{pk}_2 \leftarrow g^{\mathrm{sk}_2}$. Publish $\mathrm{pk} \leftarrow (\mathrm{pk}_1, \mathrm{pk}_2)$, set $\mathrm{sk} \leftarrow (\mathrm{sk}_1, \mathrm{sk}_2)$.
Encryption $E_{\mathrm{pk}}(m; \cdot)$. Return $\perp$ if $m \notin \mathbb{G}$. Otherwise, select a random $r \leftarrow \mathbb{Z}_q$, set $E_{\mathrm{pk}}(m; r) \leftarrow (g^r, \mathrm{pk}_1^r, m \cdot \mathrm{pk}_2^r)$.
Decryption $D_{\mathrm{sk}}(c)$. Parse $c = (c_1, c_2, c_3)$, return $\perp$ if $c_i \notin \mathbb{G}$ for some i. Return $\perp$ if $c_2 \neq c_1^{\mathrm{sk}_1}$. Otherwise, return $D_{sk}(c) \leftarrow c_3 / c_1^{\mathrm{sk}_2}$.

Let $\Pi = (G, E, D)$ be a public-key cryptosystem. The *CCA1-game* for Π is defined as follows:

Setup phase. Challenger chooses $(\mathrm{sk}, \mathrm{pk}) \leftarrow G(1^k)$ and sends pk to adversary $\mathcal{A}$.
Query phase. $\mathcal{A}$ has access to an oracle $D_{\mathrm{sk}}(\cdot)$.

Challenge phase. $\mathcal{A}$ submits (m_0, m_1) to the challenger, who picks a random bit $b_{\mathcal{A}} \leftarrow \{0,1\}$ and a random $r \leftarrow \mathbb{Z}_q$, and returns $E_{\mathrm{pk}}(m_{b_{\mathcal{A}}}; r)$.
Guess phase. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}} \in \{0,1\}$. $\mathcal{A}$ wins if $b'_{\mathcal{A}} = b_{\mathcal{A}}$.

A public-key cryptosystem is $(\tau, \gamma, \varepsilon)$-*CCA1-secure* if for any adversary $\mathcal{A}$ working in time τ and making γ queries, $\Pr[\mathcal{A}$ wins in the CCA1-game$] \leq \frac{1}{2} + \varepsilon$. A $(\tau, 0, \varepsilon)$-CCA1-secure cryptosystem is also said to be (τ, ε)-*CPA-secure*.

The DEG cryptosystem was proven to be CCA1-secure under the $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption in [Gjø06]. More precisely, Gjøsten proved the CCA1-security of a generalization of the DEG cryptosystem under a generalization of the $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption. Elgamal's cryptosystem is known to be CPA-secure [TY98] but not known to be CCA1-secure for $\gamma = \mathrm{poly}(k)$.

3 CCA1-Security of DEG

In this section we investigate the CCA1-security of DEG.

3.1 DEG Is CCA1-Secure $\Leftrightarrow$ $\mathbf{DDEG}^{\mathbf{CDEG[2]}}$

First, we prove that the security of DEG is equivalent to a new but standard-looking assumption $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ that—as one can interpret the results—basically states that DEG is CCA1-secure. This result itself is thus not so interesting, but combined with the result from the next subsection it will provide a reduction of the CCA1-security of DEG to the more standard (but as we will also see later, a likely stronger) $\mathrm{DDH}^{\mathrm{DDH}[2]}$ assumption.

THE $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ ASSUMPTION. We first define the new assumption. For implicitly defined $g, \mathrm{pk}_1, \mathrm{pk}_2$, let $\mathcal{DEG}_0 := \{(g^y, \mathrm{pk}_1^y, \mathrm{pk}_2^z) : y, z \leftarrow \mathbb{Z}_q\}$ and $\mathcal{DEG}_1 := \{(g^y, \mathrm{pk}_1^y, \mathrm{pk}_2^y) : y \leftarrow \mathbb{Z}_q\}$. Define the next oracles $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot)$ and $\mathrm{ddeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot, \cdot)$:

- $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, h_1, h_2)$ first checks if $\mathrm{ddh}(g, \mathrm{pk}_1, h_1, h_2) = 1$. If this is not true, it returns $\perp$. Otherwise, it returns $h_3 \leftarrow \mathrm{cdh}(g, \mathrm{pk}_2, h_1)$.
- $\mathrm{ddeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, h_1, h_2, h_3)$ has to distinguish between $\mathcal{DEG}_0$ and $\mathcal{DEG}_1$. That is, on the promise that $\mathrm{ddh}(g, \mathrm{pk}_1, h_1, h_2) = 1$, $\mathrm{ddeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, h_1, h_2, h_3) \leftarrow [\mathrm{ddh}(g, \mathrm{pk}_2, h_1, h_3) \overset{?}{=} 1]$. The oracle is not required to output anything if $\mathrm{ddh}(g, \mathrm{pk}_1, h_1, h_2) = 0$.

Fix a group $\mathbb{G} = \langle g \rangle$ of order q. The $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ *game* in group $\mathbb{G}$ is defined as follows:

Setup phase. Challenger sets $\mathrm{sk}_1, \mathrm{sk}_2 \leftarrow \mathbb{Z}_q$, $\mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}$, $\mathrm{pk}_2 \leftarrow g^{\mathrm{sk}_2}$. He sends $\mathrm{pk} \leftarrow (\mathrm{pk}_1, \mathrm{pk}_2)$ to adversary $\mathcal{A}$, and sets $\mathrm{sk} \leftarrow (\mathrm{sk}_1, \mathrm{sk}_2)$.
Query phase. $\mathcal{A}$ has access to the oracle $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot)$.
Challenge phase. Challenger sets $b_{\mathcal{A}} \leftarrow \{0,1\}$, $y^*, z^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$, and $h_2^* \leftarrow \mathrm{pk}_1^{y^*}$. If $b_{\mathcal{A}} = 0$, then $h_3^* \leftarrow \mathbb{G}$. If $b_{\mathcal{A}} = 1$, then $h_3^* \leftarrow \mathrm{pk}_2^{y^*}$. Challenger sends (h_1^*, h_2^*, h_3^*) to $\mathcal{A}$.
Guess phase. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}} \in \{0,1\}$. $\mathcal{A}$ wins if $b'_{\mathcal{A}} = b_{\mathcal{A}}$.

Group $\mathbb{G}$ is a $(\tau, \gamma, \varepsilon)$-$\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ *group* if for any adversary $\mathcal{A}$ working in time τ and making γ queries, $\Pr[\mathcal{A}$ wins$] \leq \frac{1}{2} + \varepsilon$. Note that this definition does directly follow from the definition of the cdeg and ddeg oracles.

SECURITY RESULTS. In all next results, small denotes some unspecified small value (usually $O(1)$ group operations) that is dominated by some other addend in the same formula.

Theorem 1 (DEG-CCA1 $\Leftrightarrow$ $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$). *Fix a group $\mathbb{G} = \langle g \rangle$ of order q.*
(1) Assume that $\mathbb{G}$ is a $(\tau, \gamma, \varepsilon)$-$\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ group. Then DEG is $(\tau - \gamma \cdot (\tau_{\mathrm{cdeg}} + \mathsf{small}) - \mathsf{small}, \gamma, 2\varepsilon)$-CCA1-secure where τ_{cdeg} is the working time of the $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot)$ oracle.
(2) Assume that DEG is $(\tau, \gamma, \varepsilon)$-CCA1-secure. Then $\mathbb{G}$ is a $(\tau - \gamma \cdot (\tau_D + \mathsf{small}) - \mathsf{small}, \gamma, \varepsilon)$-$\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ group, where τ_D is the working time of the decryption oracle D.

Proof. 1) **First direction (DEG-CCA1 $\Rightarrow$ DDEG$^{\mathrm{CDEG}[2]}$):** Assume $\mathcal{A}$ is an adversary who can $(\tau', \gamma', \varepsilon')$-break the CCA1-security of DEG with probability ε' and in time τ', making γ' queries. Construct the next reduction $\mathcal{R}$ that aims to break DDEG$^{\mathrm{CDEG}[2]}$:

- Challenger generates new $\mathrm{sk} \leftarrow (\mathrm{sk}_1, \mathrm{sk}_2) \leftarrow \mathbb{Z}_q^2$, $\mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}$, $\mathrm{pk}_2 \leftarrow g^{\mathrm{sk}_2}$ and sends $\mathrm{pk} \leftarrow (\mathrm{pk}_1, \mathrm{pk}_2)$ to $\mathcal{R}$. $\mathcal{R}$ forwards pk to $\mathcal{A}$.
- In the query phase, whenever $\mathcal{A}$ asks a decryption query (c_1, c_2, c_3), $\mathcal{R}$ rejects if either c_1, c_2 or c_3 is not a valid group element. Otherwise $\mathcal{R}$ makes a $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, c_1, c_2)$ query. $\mathcal{R}$ receives a c' such that $c' \leftarrow \perp$, if $c_2 \neq c_1^{\mathrm{sk}_1}$, and $c' \leftarrow c_1^{\mathrm{sk}_2}$ otherwise. $\mathcal{R}$ returns $\perp$ in the first case, and c_3/c' in the second case.
- In the challenge phase, whenever $\mathcal{A}$ submits her challenge (m_0^*, m_1^*), $\mathcal{R}$ asks the challenger for his own challenge. The challenger sets $b_{\mathcal{R}} \leftarrow \{0, 1\}$, $y^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$, $h_2^* \leftarrow \mathrm{pk}_1^{y^*}$. If $b_{\mathcal{R}} = 0$ then he sets $h_3^* \leftarrow \mathbb{G}$, otherwise $h_3^* \leftarrow \mathrm{pk}_2^{y^*}$. $\mathcal{R}$ picks a random bit $b_{\mathcal{A}} \leftarrow \{0, 1\}$, and sends $(h_1^*, h_2^*, m_{b_{\mathcal{A}}}^* \cdot h_3^*)$ to $\mathcal{A}$. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}}$.
- In the guess phase, if $b'_{\mathcal{A}} = b_{\mathcal{A}}$, then $\mathcal{R}$ returns $b'_{\mathcal{R}} \leftarrow 1$, otherwise $\mathcal{R}$ returns $b'_{\mathcal{R}} \leftarrow 0$.

Now,

$$\begin{aligned}\Pr[\mathcal{R}\text{ wins}] &= \Pr[b'_{\mathcal{R}} = b_{\mathcal{R}}] = \Pr[\mathcal{A}\text{ wins}|b_{\mathcal{R}} = 1] \cdot \Pr[b_{\mathcal{R}} = 1] + \Pr[\mathcal{A}\text{ wins}|b_{\mathcal{R}} = 0] \cdot \Pr[b_{\mathcal{R}} = 0] \\ &= \left(\frac{1}{2} + \varepsilon'\right) \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\varepsilon'}{2} \ .\end{aligned}$$

Clearly $\mathcal{R}$ works in time $\tau = \tau' + \gamma \cdot (\tau_{\mathrm{cdeg}} + \mathsf{small}) + \mathsf{small}$. □

2) **Second direction** (DDEG$^{\mathrm{CDEG}[2]}$ $\Rightarrow$ **DEG-CCA1):** Assume $\mathcal{A}$ is an adversary who can $(\tau', \gamma', \varepsilon')$-break the DDEG$^{\mathrm{CDEG}[2]}$ assumption. Construct the next reduction $\mathcal{R}$ that aims to break the CCA1-security of the DEG cryptosystem:

- Challenger generates new $\mathrm{sk} \leftarrow (\mathrm{sk}_1, \mathrm{sk}_2) \leftarrow \mathbb{Z}_q^2$, $\mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}$, $\mathrm{pk}_2 \leftarrow g^{\mathrm{sk}_2}$, and sends $\mathrm{pk} = (\mathrm{pk}_1, \mathrm{pk}_2)$ to $\mathcal{R}$. $\mathcal{R}$ forwards pk to $\mathcal{A}$.
- In the query phase, whenever $\mathcal{A}$ asks a query $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, h_1, h_2)$, $\mathcal{R}$ makes a decryption query $(h_1, h_2, 1)$, and receives back either $\perp$ or $k \leftarrow h_1^{-\mathrm{sk}_2}$. $\mathcal{R}$ returns $h_3 \leftarrow \perp$ in the first case, and $h_3 \leftarrow k^{-1}$ in the second case.
- In the challenge phase, whenever $\mathcal{A}$ asks for a challenge, $\mathcal{R}$ sends his challenge pair $(m_0^*, m_1^*) \leftarrow (g^{r_1^*}, 1)$, for $r_1^* \leftarrow \mathbb{Z}_q$, to the challenger. Challenger picks a random bit $b_{\mathcal{R}} \leftarrow \{0, 1\}$ and a random $r_2^* \leftarrow \mathbb{Z}_q$, and sends $(c_1^*, c_2^*, c_3^*) \leftarrow (g^{r_2^*}, \mathrm{pk}_1^{r_2^*}, g^{r_1^*(1-b_{\mathcal{R}})} \cdot \mathrm{pk}_2^{r_2^*})$ to $\mathcal{R}$. $\mathcal{R}$ forwards (c_1^*, c_2^*, c_3^*) to $\mathcal{A}$, who returns a guess $b'_{\mathcal{A}}$.
- In the guess phase, $\mathcal{R}$ returns $b'_{\mathcal{R}} \leftarrow b'_{\mathcal{A}}$ to challenger.

Now, $\Pr[\mathcal{R}\text{ wins}] = \Pr[b'_{\mathcal{R}} = b_{\mathcal{R}}] = \Pr[\mathcal{A}\text{ wins}] = \varepsilon'$. Clearly $\mathcal{R}$ works in time $\tau' + \gamma \cdot (\tau_D + \mathsf{small}) + \mathsf{small}$. □

The next theorem and its corollary show that DEG's CCA1-security can be based on a more traditional (albeit a stronger) assumption. The proof is a hybrid argument of the type we already mentioned in Sect. 2.2 but we will bring it here for the sake of completeness. In the proof we use the trapdoor test trick from [CKS08].

Theorem 2 (DDEG$^{\mathrm{CDEG}[2]}$ $\Rightarrow$ DDH$^{\mathrm{DDH}[2]}$).
1) Any $(\tau, \gamma, \varepsilon)$-DDEG$^{\mathrm{DDH}[2]}$ group $\mathbb{G} = \langle g\rangle$ is also a $(\tau - \gamma \cdot (\tau_{\mathrm{cdeg}} + \mathsf{small}) - \mathsf{small}, \gamma, \varepsilon)$-DDEG$^{\mathrm{CDEG}[2]}$ group, where τ_{cdeg} is the working time of the $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot, \cdot)$ oracle and the ddh oracle has parameters $\mathrm{ddh}(g, \mathrm{pk}_2, \cdot, \cdot)$.
2) Any $(\tau, \gamma, \varepsilon)$-DDH$^{\mathrm{DDH}[2]}$ group $\mathbb{G} = \langle g\rangle$ is also a $(\tau - \gamma \cdot (\tau_{\mathrm{ddh}} + \mathsf{small}) - \mathsf{small}, \gamma, \varepsilon)$-DDEG$^{\mathrm{DDH}[2]}$ group, where τ_{ddh} is the working time of the $\mathrm{ddh}(g, \mathrm{pk}_2, \cdot, \cdot)$ oracle and the ddh oracle has parameters $\mathrm{ddh}(g, \mathrm{pk}_2, \cdot, \cdot)$.

Proof. 1) **First claim** (DDEG$^{\mathrm{CDEG}[2]}$ $\Rightarrow$ DDEG$^{\mathrm{DDH}[2]}$)**:** Fix a group $\mathbb{G} = \langle g\rangle$ of order q. Assume $\mathcal{A}$ is an adversary who can $(\tau', \gamma', \varepsilon')$-break the DDEG$^{\mathrm{CDEG}[2]}$ assumption. Construct the next reduction $\mathcal{R}$ that aims to break DDEG$^{\mathrm{DDH}[2]}$ in the same group:

- Challenger generates new $(\mathrm{sk} \leftarrow \mathbb{Z}_q, \mathrm{pk} \leftarrow g^{\mathrm{sk}})$, and sends pk to $\mathcal{R}$. $\mathcal{R}$ sets $\mathrm{pk}_1 \leftarrow \mathrm{pk}$, $u \leftarrow \mathbb{Z}_q$, $v \leftarrow \mathbb{Z}_q$, $\mathrm{pk}_2 \leftarrow g^u / \mathrm{pk}_1^v$. Thus $\mathrm{pk}_2 = g^{\mathrm{sk}_2}$ for $\mathrm{sk}_2 \leftarrow u - v \cdot \mathrm{sk}_1$. He forwards $(\mathrm{pk}_1, \mathrm{pk}_2)$ as the public key to $\mathcal{A}$.

- In the query phase, whenever $\mathcal{A}$ asks a query $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, h_1, h_2)$, $\mathcal{R}$ makes a $\mathrm{ddh}(g, \mathrm{pk}, h_1, h_2)$ query. If this query returns 0, then $\mathcal{R}$ returns $\perp$. Otherwise, $\mathcal{R}$ computes $h_3 \leftarrow h_1^u / h_2^v$ and outputs h_3. Note that in this case $h_3 = h_1^u / h_2^v = h_1^u / h_1^{v \cdot \mathrm{sk}_1} = h_1^{\mathrm{sk}_2}$, and thus $\mathcal{A}$ gets the correct output.
- In the challenge phase, if $\mathcal{A}$ asks for a challenge from $\mathcal{R}$, then $\mathcal{R}$ asks for a challenge from the challenger. Challenger sets $b_{\mathcal{R}} \leftarrow \{0,1\}$, $y^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$, $h_2^* \leftarrow \mathbb{G}$. If $b_{\mathcal{R}} = 0$ then he sets $h_3^* \leftarrow \mathbb{G}$, otherwise $h_3^* \leftarrow \mathrm{pk}_2^{y^*}$. Challenger sends (h_1^*, h_2^*, h_3^*) to $\mathcal{R}$. $\mathcal{R}$ sends (h_1^*, h_2^*, h_3^*) to $\mathcal{A}$. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}}$.
- In the guess phase, $\mathcal{R}$ returns $b'_{\mathcal{R}} \leftarrow b'_{\mathcal{A}}$.

Clearly, $\mathcal{R}$ wins if and only if $\mathcal{A}$ wins.
2) **Second claim** ($\mathrm{DDEG}^{\mathrm{DDH}[2]} \Rightarrow \mathrm{DDH}^{\mathrm{DDH}[2]}$)**:** Fix a group $\mathbb{G} = \langle g \rangle$ of order q. Assume $\mathcal{A}$ is an adversary who can $(\tau', \gamma', \varepsilon')$-break the $\mathrm{DDEG}^{\mathrm{DDH}[2]}$ assumption. Construct the next reduction $\mathcal{R}$ that aims to break $\mathrm{DDH}^{\mathrm{DDH}[2]}$ in the same group:

- Challenger generates new $(\mathrm{sk} \leftarrow \mathbb{Z}_q, \mathrm{pk} \leftarrow g^{\mathrm{sk}})$, and sends pk to $\mathcal{R}$. $\mathcal{R}$ sets $\mathrm{pk}_2 \leftarrow \mathrm{pk}$, $\mathrm{sk}_1 \leftarrow \mathbb{Z}_q$, $\mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}$. He forwards $(\mathrm{pk}_1, \mathrm{pk}_2)$ as the public key to $\mathcal{A}$.
- In the query phase, whenever $\mathcal{A}$ asks a query $\mathrm{ddh}(g, \mathrm{pk}_2, h_1, h_2)$, $\mathcal{R}$ makes the same query and forwards the reply to $\mathcal{R}$.
- In the challenge phase, if $\mathcal{A}$ asks for a challenge from $\mathcal{R}$, then $\mathcal{R}$ asks for a challenge from the challenger. Challenger sets $b_{\mathcal{R}} \leftarrow \{0,1\}$, $y^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$. If $b_{\mathcal{R}} = 0$ then he sets $h_2^* \leftarrow \mathbb{G}$, otherwise $h_2^* \leftarrow \mathrm{pk}_2^{y^*}$. Challenger sends (h_1^*, h_2^*) to $\mathcal{R}$. $\mathcal{R}$ sets $h_3^* \leftarrow (h_1^*)^{\mathrm{pk}_1}$. $\mathcal{R}$ sends (h_1^*, h_3^*, h_2^*) to $\mathcal{A}$. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}}$.
- In the guess phase, $\mathcal{R}$ returns $b'_{\mathcal{R}} \leftarrow b'_{\mathcal{A}}$.

Clearly, $\mathcal{R}$ wins if and only if $\mathcal{A}$ wins. □

Corollary 1 (DEG-CCA1 $\Rightarrow$ $\mathrm{DDH}^{\mathrm{DDH}[2]}$). *Assume that $\mathbb{G} = \langle g \rangle$ is a $(\tau, \gamma, \varepsilon)$-$\mathrm{DDH}^{\mathrm{DDH}[2]}$ group. Then the DEG cryptosystem is CCA1-secure in group $\mathbb{G}$.*

By following a very similar proof, a variant of the DEG cryptosystem where the decryption, given an invalid ciphertext, returns a random plaintext instead of $\perp$, is CCA1-secure under the DDH assumption.

RELATION WITH DDH.

Theorem 3 ($\mathrm{DDH} \Rightarrow \mathrm{DDEG}^{\mathrm{CDEG}[2]}$)**.** *Any $(\tau, \gamma, \varepsilon)$-$\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ group $\mathbb{G} = \langle g \rangle$ is also a $(\tau - \mathsf{small}, \varepsilon)$-DDH group.*

Proof. Fix a group $\mathbb{G} = \langle g \rangle$ of order q. Assume $\mathcal{A}$ is an adversary who can $(\tau', \gamma', \varepsilon')$-break the DDH assumption. Construct the next reduction $\mathcal{R}$ that aims to break $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ in the same group:

- Challenger generates new $(\mathrm{sk}_1 \leftarrow \mathbb{Z}_q, \mathrm{sk}_2 \leftarrow \mathbb{Z}_q, \mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}, \mathrm{pk}_2 \leftarrow g^{\mathrm{sk}_2})$ and sends $\mathrm{pk} = (\mathrm{pk}_1, \mathrm{pk}_2)$ to $\mathcal{R}$. $\mathcal{R}$ forwards (g, pk_2) to $\mathcal{A}$ as her system parameters.
- In the challenge phase, if $\mathcal{A}$ asks for a challenge then $\mathcal{R}$ asks for a challenge. Challenger sets $b_{\mathcal{R}} \leftarrow \{0,1\}$, $y^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$, $h_2^* \leftarrow \mathrm{pk}_1^{y^*}$. If $b_{\mathcal{R}} = 0$ then he sets $h_3^* \leftarrow \mathbb{G}$, otherwise $h_3^* \leftarrow \mathrm{pk}_2^{y^*}$. He sends (h_1^*, h_2^*, h_3^*) to $\mathcal{R}$. $\mathcal{R}$ sends (h_1^*, h_3^*) to $\mathcal{A}$. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}}$. $\mathcal{R}$ returns $b'_{\mathcal{R}} \leftarrow b'_{\mathcal{A}}$ to the challenger.

Clearly, $\mathcal{R}$ wins if and only if $\mathcal{A}$ wins. □

4 CCA1-Security of ElGamal

To prove the security of ElGamal we need the next assumption. As we will see from the security proofs, this assumption basically just asserts that Elgamal is CCA1-secure.

Fix a group $\mathbb{G} = \langle g \rangle$ of order q. The $\mathrm{DDH}^{\mathrm{CDH}[1]}$ *game* is defined as follows:

Setup phase. Challenger sets $\mathrm{sk} \leftarrow \mathbb{Z}_q$, $\mathrm{pk} \leftarrow g^{\mathrm{sk}}$. He sends pk to adversary $\mathcal{A}$.

Query phase. $\mathcal{A}$ has access to oracle $\mathrm{cdh}(g, \mathrm{pk}, \cdot)$, that is, $\mathrm{cdh}(g, \mathrm{pk}, h) := h^{\mathrm{sk}}$.
Challenge phase. Challenger sets $b_{\mathcal{A}} \leftarrow \{0,1\}$, $y^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$. He sets $h_2^* \leftarrow \mathbb{G}$ if $b_{\mathcal{A}} = 0$ and $h_2^* \leftarrow \mathrm{pk}^{y^*} = \mathrm{cdh}(g, \mathrm{pk}, h_1^*)$ if $b_{\mathcal{A}} = 1$. Challenger sends (h_1^*, h_2^*) to $\mathcal{A}$.
Guess phase. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}} \in \{0,1\}$. $\mathcal{A}$ wins if $b'_{\mathcal{A}} = b_{\mathcal{A}}$, that is, if $b_{\mathcal{A}} = \mathrm{ddh}(g, \mathrm{pk}, h_1^*, h_2^*)$.

Group $\mathbb{G}$ is a $(\tau, \gamma, \varepsilon)$-$\mathrm{DDH}^{\mathrm{CDH}[1]}$ *group* if for any adversary $\mathcal{A}$ working in time τ and making γ queries, $\Pr[\mathcal{A}$ wins in the $\mathrm{DDH}^{\mathrm{CDH}[1]}$ game$] \leq \frac{1}{2} + \varepsilon$.

Theorem 4 (Elgamal-CCA1 $\Leftrightarrow$ $\mathrm{DDH}^{\mathrm{CDH}[1]}$). *Fix a group $\mathbb{G} = \langle g \rangle$ of order q.*
(1) Assume that $\mathbb{G}$ is a $(\tau, \gamma, \varepsilon)$-$\mathrm{DDH}^{\mathrm{CDH}[1]}$ group. Then ElGamal is $(\tau - \gamma \cdot (\tau_{\mathrm{cdh}} + \mathsf{small}) - \mathsf{small}, \gamma, 2\varepsilon)$-CCA1-secure, where τ_{cdh} is the working time of the $\mathrm{cdh}(g, \mathrm{pk}, \cdot)$ oracle.
(2) Assume that ElGamal is $(\tau, \gamma, \varepsilon)$-CCA1-secure. Then $\mathbb{G}$ is a $(\tau - \gamma \cdot (\tau_D + \mathsf{small}) - \mathsf{small}, \gamma, \varepsilon)$-$\mathrm{DDH}^{\mathrm{CDH}[1]}$ group, where τ_D is the working time of the D oracle.

Proof. 1) **First direction (Elgamal-CCA1 $\Rightarrow$ $\mathrm{DDH}^{\mathrm{CDH}[1]}$):** Assume $\mathcal{A}$ is an adversary who can $(\tau', \gamma', \varepsilon')$-break the CCA1-security of Elgamal in group $\mathbb{G}$ with probability ε' and in time τ', making γ' queries. Construct the next reduction $\mathcal{R}$ that aims to break $\mathrm{DDH}^{\mathrm{CDH}[1]}$ in group $\mathbb{G}$:

- Challenger generates a new keypair ($\mathrm{sk} \leftarrow \mathbb{Z}_q, \mathrm{pk} \leftarrow g^{\mathrm{sk}}$) and sends pk to $\mathcal{R}$. $\mathcal{R}$ forwards pk to $\mathcal{A}$.
- In the query phase, whenever $\mathcal{A}$ asks a decryption query (c_1, c_2), $\mathcal{R}$ rejects if either c_1 or c_2 is not a valid group element. Otherwise $\mathcal{R}$ asks a CDH query $c_3 \leftarrow \mathrm{cdh}(g, \mathrm{pk}, c_1)$. $\mathcal{R}$ returns c_2/c_3.
- In the challenge phase, whenever $\mathcal{A}$ gives a pair (m_0^*, m_1^*) of messages, $\mathcal{R}$ asks his challenge from the challenger. The challenger sets $b_{\mathcal{R}} \leftarrow \{0,1\}$, $y^* \leftarrow \mathbb{Z}_q$, $h_1^* \leftarrow g^{y^*}$. If $b_{\mathcal{R}} = 0$ then he sets $h_2^* \leftarrow \mathbb{G}$, otherwise $h_2^* \leftarrow \mathrm{pk}^{y^*}$. $\mathcal{R}$ picks a random bit $b_{\mathcal{A}} \leftarrow \{0,1\}$ and sends $(h_1, m_{b_{\mathcal{A}}} \cdot h_2)$ to $\mathcal{A}$. $\mathcal{A}$ returns a bit $b'_{\mathcal{A}}$.
- In the guess phase, if $b'_{\mathcal{A}} = b_{\mathcal{A}}$ then $\mathcal{R}$ returns $b'_{\mathcal{R}} = 1$, otherwise $\mathcal{R}$ returns $b'_{\mathcal{R}} = 0$.

Now, $\Pr[\mathcal{R}$ wins in the $\mathrm{DDH}^{\mathrm{CDH}[1]}$ game$] = \Pr[b'_{\mathcal{R}} = b_{\mathcal{R}}] = \Pr[\mathcal{A} \text{ wins} | b_{\mathcal{R}} = 1] \cdot \Pr[b_{\mathcal{R}} = 1] + \Pr[\mathcal{A} \text{ wins} | b_{\mathcal{R}} = 0] \cdot \Pr[b_{\mathcal{R}} = 0] = (\frac{1}{2} + \varepsilon') \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\varepsilon'}{2}$. Clearly $\mathcal{R}$ works in time $\tau' + \gamma \cdot (\tau_{\mathrm{cdh}} + \mathsf{small}) + \mathsf{small}$. □

2) **Second direction** ($\mathrm{DDH}^{\mathrm{CDH}[1]}$ $\Rightarrow$ **Elgamal-CCA1):** Assume $\mathcal{A}$ is an adversary who can $(\tau', \gamma', \varepsilon')$-break the $\mathrm{DDH}^{\mathrm{CDH}[1]}$ assumption in group $\mathbb{G}$. Construct the next reduction $\mathcal{R}$ that aims to break the CCA1-security of Elgamal:

- Challenger generates a new keypair ($\mathrm{sk} \leftarrow \mathbb{Z}_q, \mathrm{pk} \leftarrow g^{\mathrm{sk}}$) and sends pk to $\mathcal{R}$. $\mathcal{R}$ forwards pk to $\mathcal{A}$.
- In the query phase, whenever $\mathcal{A}$ asks a CDH query $\mathrm{cdh}(g, \mathrm{pk}, h)$, $\mathcal{R}$ asks a decryption query $(h, 1)$, and receives back $c \leftarrow h^{-\mathrm{sk}}$. $\mathcal{R}$ returns c^{-1} to $\mathcal{A}$.
- In the challenge phase, whenever $\mathcal{A}$ asks for a challenge, $\mathcal{R}$ sends his challenge $(m_0^*, m_1^*) \leftarrow (g, 1)$ to challenger. Challenger pick a random bit $b_{\mathcal{R}} \leftarrow \{0,1\}$ and a random $r^* \leftarrow \mathbb{Z}_q$, and sends $(c_1^*, c_2^*) \leftarrow (g^{r^*}, g^{1-b_{\mathcal{R}}} \cdot \mathrm{pk}^{r^*})$ to $\mathcal{R}$. $\mathcal{R}$ forwards (c_1^*, c_2^*) to $\mathcal{A}$, who returns a guess $b'_{\mathcal{A}}$. $\mathcal{R}$ returns $b'_{\mathcal{R}} \leftarrow b'_{\mathcal{A}}$ to challenger.

Now, $\Pr[\mathcal{R} \text{ wins}] = \Pr[b'_{\mathcal{R}} = b_{\mathcal{R}}] = \Pr[\mathcal{A} \text{ wins}] = \varepsilon$. Clearly $\mathcal{R}$ works in time $\tau' + \gamma \cdot (\tau_D + \mathsf{small}) + \mathsf{small}$. □

5 Irreductions

IRREDUCTIONS IN THE GENERIC GROUP MODEL. The $\mathrm{DDH}^{\mathrm{CDH}[1]}$ assumption is a direct opposite of the well-known StrongCDH assumption [OP01] where one has initial access to the DDH oracle and then has to compute CDH. Thus, $\mathrm{DDH}^{\mathrm{CDH}[1]}$ is probably stronger than the StrongDH assumption. We can formally prove the next result:

Theorem 5. *In the generic group model where one also knows the factorization of $q - 1$ for group order q,* $\mathrm{DDH}^{\mathrm{CDH}[1]} \not\Rightarrow \mathrm{DDH}$.

Proof. According to [Che06], if $q-1$ has a polynomially small divisor d then sk can be found from (g, g^{sk^d}) in $O(\log q \cdot (\sqrt{(q-1)/d} + \sqrt{d}))$ group operations and with $\Theta(\sqrt{p/d} + \sqrt{d})$ memory. Because only the knowledge of (g, g^{sk^d}) is necessary, the adversary can obtain g^{sk^d} by querying the group oracle $\Theta(\log d)$ times with inputs $\mathrm{pk}^{msb(d,i)}$, where $msb(d,i)$ denotes the i most significant bits of d. (Recall that [Che06] said that the oracle has to be applied d times!)

On the other hand, one needs $\Theta(\log q \cdot \sqrt{q})$ group operations in the generic group model [Sho97] to solve the DDH assumption. For d, superpolylogarithmic in the security parameter, $\mathrm{DDH}^{\mathrm{CDH}[1]}$ is thus strictly stronger than DDH in the generic group model. □

What this theorem does say is that there does not exist a polynomial-time reduction $\mathrm{DDH} \Rightarrow \mathrm{DDH}^{\mathrm{CDH}[1]}$ that uses group operations only: if such a reduction existed, it could be used to derive an algorithm to compute DDH in the generic group model with the same complexity as stated in this theorem. This would be in contradiction with [Sho97]. Moreover, there may exist groups where $\mathrm{DDH} \not\Rightarrow \mathrm{DDH}^{\mathrm{CDH}[1]}$. See [Che06] for example elliptic curve groups where currently $\mathrm{DDH}^{\mathrm{CDH}[1]}$ seems to be much easier than DDH. In the extreme when $d = \Theta(\sqrt{q})$, in the generic group model $\mathrm{DDH}^{\mathrm{CDH}[1]}$ can be solved in $\Theta(\log q \cdot \sqrt[4]{q})$ group operations while DDH can be solved in $\Theta(\log q \cdot \sqrt{q})$ group operations. (Cheon also provides an algorithm for the case when $q+1$ has known factors [Che06].)

CONDITIONAL IRREDUCTIONS. In what follows, we will not state the concrete security parameters in the theorems, however, they are easy to calculate and one can verify that all following theorems provide exact (ir)reductions.

Theorem 6 ($\mathrm{DDEG}^{\mathrm{CDEG}[2]} \not\Rightarrow_! \mathrm{DDH}$)**.** *If there is a reduction $\mathcal{R}$ that reduces* DDH *to* $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$*, then there is an efficient irreduction $\mathcal{I}$ that, given $\mathcal{R}$ as an oracle, solves* DDH.

Proof. Fix a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order q. Let $\mathcal{A} = \mathcal{A}^{\mathrm{cdeg}}$, be an arbitrary algorithm that solves $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$. Assume that $\mathcal{R} = \mathcal{R}^{\mathcal{A}}$ is an arbitrary reduction that uses $\mathcal{A}$ as an oracle to solve DDH. Construct now the next oracle machine $\mathcal{I} = \mathcal{I}^{\mathcal{R}}$ to solve DDH, in time and with success probability comparable with those of $\mathcal{R}$.

- Challenger sets $\mathrm{sk}_1, \mathrm{sk}_2 \in \mathbb{Z}_q$ and $(\mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}, \mathrm{pk}_2 \leftarrow g^{\mathrm{sk}_2})$. He sends $\mathrm{pk} \leftarrow (\mathrm{pk}_1, \mathrm{pk}_2)$ to $\mathcal{I}$ as the public key. $\mathcal{I}$ forwards pk to $\mathcal{R}$.
- In the challenge phase, whenever $\mathcal{R}$ asks for a challenge from $\mathcal{I}$, $\mathcal{I}$ asks for a challenge from challenger. Challenger sets $b_{\mathcal{I}} \leftarrow \{0,1\}$ and $y^* \leftarrow \mathbb{Z}_q$. He sets $h_1^* \leftarrow g^{y^*}$. If $b_{\mathcal{I}} = 0$ then he sets $h_2^* \leftarrow \mathbb{G}$, otherwise he sets $h_2^* \leftarrow \mathrm{pk}^{y^*} = \mathrm{cdh}(g, \mathrm{pk}, h_1^*)$. The challenger sends (h_1^*, h_2^*) to $\mathcal{I}$ as a challenge. $\mathcal{I}$ sends (h_1^*, h_3^*) to $\mathcal{R}$ as his challenge in the $\mathrm{DDEG}^{\mathrm{CDEG}[2]} \Rightarrow \mathrm{DDH}$ game. Note that $b_{\mathcal{R}} = b_{\mathcal{I}}$. She then simulates challenger and $\mathcal{A}$ to $\mathcal{R}$ in the $\mathrm{DDEG}^{\mathrm{CDEG}[2]} \Rightarrow \mathrm{DDH}$ game as follows:
  - Whenever $\mathcal{R}$ asks a $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ query $(g, \mathrm{pk}_1, \mathrm{pk}_2, h_1, h_2, h_3)$ from $\mathcal{A}$, $\mathcal{I}$ simulates $\mathcal{A}$ as follows:
    1. $\mathcal{I}$ uses the random self-reducibility property of DDH to transform (h_1, h_2) to $(h_1' \leftarrow h_1 \cdot g^z, h_2' \leftarrow h_2 \cdot \mathrm{pk}_1^z)$, for $z \leftarrow \mathbb{Z}_q$, and sends (h_1', h_2') to $\mathcal{R}$ as a $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot)$ query in the $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ game. $\mathcal{R}$ replies with some h_3'.
    2. If $h_3' = h_3 \cdot \mathrm{pk}_2^z$ then $b_{\mathcal{A}}' \leftarrow 1$, otherwise $b_{\mathcal{A}}' \leftarrow 0$. $\mathcal{I}$ replies with $b_{\mathcal{A}}'$ as her answer to the $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ challenge.
  - In the challenge phase, after some γ queries, $\mathcal{R}$ outputs a bit $b_{\mathcal{R}}'$. $\mathcal{I}$ returns $b_{\mathcal{R}}'$.

First, $\mathcal{I}$ emulates $\mathcal{A}$ correctly. Moreover, if $\mathcal{R}$ responds incorrectly to the DDH query then $\mathcal{A} = \mathcal{I}$ is not required to answer correctly. Thus, $\Pr[\mathcal{I} \text{ wins}] = \Pr[\mathcal{R} \text{ wins}]$, and $\mathcal{I}$ spends marginally more time than $\mathcal{R}$. □

Theorem 7 ($\mathrm{DDH}^{\mathrm{DDH}[2]} \not\Rightarrow_! \mathrm{DDEG}^{\mathrm{CDEG}[2]}$)**.** *If there is a reduction $\mathcal{R}$ that reduces* $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$ *to* $\mathrm{DDH}^{\mathrm{DDH}[2]}$*, then there is an efficient irreduction $\mathcal{I}$ that, given $\mathcal{R}$ as an oracle, solves* $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$.

Proof. Fix a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order q. Let $\mathcal{A} = \mathcal{A}^{\mathrm{ddh}}$ be an arbitrary algorithm that solves $\mathrm{DDH}^{\mathrm{DDH}[2]}$. Assume that $\mathcal{R} = \mathcal{R}^{\mathcal{A},\mathrm{cdeg}}$ is an efficient algorithm that uses $\mathcal{A}$ as an oracle to solve $\mathrm{DDEG}^{\mathrm{CDEG}[2]}$. Construct now the next oracle machine $\mathcal{I} = \mathcal{I}^{\mathcal{R},\mathrm{cdeg}}$ to solve DDEG with the help of $\mathcal{R}$ and $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot)$ as oracles, in time and with success probability comparable with those of $\mathcal{R}$.

- Challenger sets $\mathrm{sk}_1, \mathrm{sk}_2 \in \mathbb{Z}_q$ and $\mathrm{pk} \leftarrow (\mathrm{pk}_1 \leftarrow g^{\mathrm{sk}_1}, \mathrm{pk}_2 \leftarrow g^{\mathrm{sk}_2})$. He sends pk to $\mathcal{I}$ as the public key. $\mathcal{I}$ forwards pk to $\mathcal{R}$ as his public key.
- Query phase of the $\mathrm{DDH}^{\mathrm{DDH}[2]} \Rightarrow \mathrm{DDEG}^{\mathrm{CDEG}[2]}$ game:
  - If $\mathcal{R}$ asks a $\mathrm{cdeg}(g, \mathrm{pk}_1, \mathrm{pk}_2, \cdot, \cdot)$ query (h_1, h_2) from $\mathcal{A}$, $\mathcal{I}$ forwards it to her own cdeg oracle.
  - Whenever $\mathcal{R}$ asks a $\mathrm{DDH}^{\mathrm{DDH}[2]}$ query (h_1, h_2) from $\mathcal{A}$, $\mathcal{I}$ uses the random self-reducibility property of DDH to transform (h_1, h_2) to $(h_1' \leftarrow h_1 \cdot g^z, h_2' \leftarrow h_2 \cdot \mathrm{pk}_1^z)$, for random $z \leftarrow \mathbb{Z}_q$, and sends (h_1', h_2') to the DDH oracle ($\mathcal{R}$). When $\mathcal{R}$ returns b'_{ddh}, $\mathcal{I}$ returns $b'_{\mathcal{A}} \leftarrow b'_{\mathrm{ddh}}$.
- In the challenge phase, whenever $\mathcal{R}$ asks for a DDEG challenge from $\mathcal{I}$, $\mathcal{I}$ asks for a challenge from challenger. Challenger sets $b_{\mathcal{I}} \leftarrow \{0, 1\}$ and $y^* \leftarrow \mathbb{Z}_q$. He sets $h_1^* \leftarrow g^{y^*}$ and $h_2^* \leftarrow \mathrm{pk}_1^{y^*}$. If $b_{\mathcal{I}} = 0$ then he sets $h_3^* \leftarrow \mathbb{G}$, otherwise he sets $h_3^* \leftarrow \mathrm{pk}_2^{y^*}$. The challenger sends (h_1^*, h_2^*, h_3^*) to $\mathcal{I}$ as a challenge. $\mathcal{I}$ forwards (h_1^*, h_2^*, h_3^*) to $\mathcal{R}$ as his challenge.
- In the guess phase, when $\mathcal{R}$ outputs a bit $b'_{\mathcal{R}}$. $\mathcal{I}$ returns $b'_{\mathcal{I}} \leftarrow b'_{\mathcal{R}}$.

First, $\mathcal{I}$ emulates the queries correctly. Thus if $\mathcal{R}$ responds with a correct answer to the DDH query, then $\mathcal{I}$ responds with a correct answer to the $\mathrm{DDH}^{\mathrm{DDH}[2]}$ query. Thus $\Pr[\mathcal{I}\ \mathrm{wins}] = \Pr[\mathcal{R}\ \mathrm{wins}]$, and $\mathcal{I}$ works in time $\tau + \gamma_{\mathrm{ddeg}} \cdot (\tau_{\mathrm{ddeg}} + \mathsf{small}) + \gamma_{\mathcal{A}} \cdot \mathsf{small} + \mathsf{small}$, where τ is the working time of $\mathcal{R}$, τ_{ddeg} is the working time of the ddeg oracle, γ_{ddeg} is the number of queries to the ddeg oracle, $\gamma_{\mathcal{A}}$ is the number of queries to $\mathcal{A}$. □

Theorem 8 ($\mathrm{DDH}^{\mathrm{CDH}[1]} \not\Rightarrow_! \mathrm{DDH}^{\mathrm{DDH}[2]}$). *If there is a reduction $\mathcal{R}$ that reduces $\mathrm{DDH}^{\mathrm{DDH}[2]}$ to $\mathrm{DDH}^{\mathrm{CDH}[1]}$, then there is an efficient irreduction $\mathcal{I}$ that, given $\mathcal{R}$ as an oracle, solves $\mathrm{DDH}^{\mathrm{DDH}[2]}$.*

Proof. Fix a cyclic group $\mathbb{G} = \langle g \rangle$ of prime order q. Let $\mathcal{A} = \mathcal{A}^{\mathrm{cdh}}$ be an arbitrary algorithm that solves $\mathrm{DDH}^{\mathrm{CDH}[1]}$. Assume that $\mathcal{R} = \mathcal{R}^{\mathcal{A}, \mathrm{ddh}(g, \mathrm{pk}, \cdot)}$ is an efficient algorithm that uses $\mathcal{A}$ as an oracle to solve $\mathrm{DDH}^{\mathrm{DDH}[2]}$. Construct now the next oracle machine $\mathcal{I} = \mathcal{I}^{\mathcal{R}, \mathrm{ddh}}$ to solve DDH with the help of $\mathcal{R}$ and $\mathrm{ddh}(g, \mathrm{pk}, \cdot)$ as oracles, in time and with success probability comparable with those of $\mathcal{R}$.

- Challenger sets $\mathrm{sk} \in \mathbb{Z}_q$ and $\mathrm{pk} \leftarrow g^{\mathrm{sk}}$. He sends pk to $\mathcal{I}$ as the public key. $\mathcal{I}$ forwards it to $\mathcal{R}$ as his public key.
- In the query phase of the $\mathrm{DDH}^{\mathrm{CDH}[1]} \Rightarrow \mathrm{DDH}^{\mathrm{DDH}[2]}$ game:
  - If $\mathcal{R}$ asks a $\mathrm{ddh}(g, \mathrm{pk}, \cdot, \cdot)$ query (h_1, h_2) from $\mathcal{A}$, $\mathcal{I}$ forwards it to her ddh oracle.
  - If $\mathcal{R}$ asks a $\mathrm{DDH}^{\mathrm{CDH}[1]}$ query (h_1, h_2) from $\mathcal{A}$, $\mathcal{I}$ uses the random self-reducibility property of CDH to transform h_1 to $h_1' \leftarrow h_1 \cdot g^z$, for random $z \leftarrow \mathbb{Z}_q$, and sends h_1' to the CDH oracle ($\mathcal{R}$). $\mathcal{R}$ returns h'_{cdh}. If $h'_{\mathrm{cdh}} = h_2 \cdot \mathrm{pk}^z$ then $\mathcal{I}$ answers $b'_{\mathcal{A}} \leftarrow 1$, otherwise $\mathcal{I}$ answers $b'_{\mathcal{A}} \leftarrow 0$.
- In the challenge phase, when $\mathcal{R}$ asks his challenge from $\mathcal{I}$, $\mathcal{I}$ asks her challenge from the challenger. challenger sets $b_{\mathcal{I}} \leftarrow \{0, 1\}$ and $y^* \leftarrow \mathbb{Z}_q$. He sets $h_1^* \leftarrow g^{y^*}$. If $b_{\mathcal{I}} = 0$ then he sets $h_2^* \leftarrow \mathbb{G}$, otherwise he sets $h_2^* \leftarrow \mathrm{pk}^{y^*} = \mathrm{cdh}(g, \mathrm{pk}, h_1^*)$. The challenger sends (h_1^*, h_2^*) to $\mathcal{I}$ as a challenge. $\mathcal{I}$ forwards (h_1^*, h_2^*) to $\mathcal{R}$ as his challenge. $\mathcal{R}$ outputs a bit $b'_{\mathcal{R}}$.
- In the guess phase, $\mathcal{I}$ returns $b'_{\mathcal{I}} \leftarrow b'_{\mathcal{R}}$.

First, $\mathcal{I}$ emulates the queries correctly. Thus if $\mathcal{R}$ responds with a correct answer to the CDH query, then $\mathcal{I}$ responds with a correct answer to the $\mathrm{DDH}^{\mathrm{CDH}[2]}$ query. Thus $\Pr[\mathcal{I}\ \mathrm{wins}] = \Pr[\mathcal{R}\ \mathrm{wins}]$, and $\mathcal{I}$ works in time $\tau + \gamma_{\mathrm{ddh}} \cdot (\tau_{\mathrm{ddh}} + \mathsf{small}) + \gamma_{\mathcal{A}} \cdot \mathsf{small} + \mathsf{small}$, where τ is the working time of $\mathcal{R}$, τ_{ddh} is the working time of the ddh oracle, γ_{ddh} is the number of queries to the ddh oracle, $\gamma_{\mathcal{A}}$ is the number of queries to $\mathcal{A}$. □

ACKNOWLEDGMENTS. The author was supported by Estonian Science Foundation, grant #6848, European Union through the European Regional Development Fund and the 6th Framework Programme project AEOLUS (FP6-IST-15964). We thank Daniel Brown, Eike Kiltz and Kenny Paterson for discussions.

References

ABR01. Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The Oracle Diffie-Hellman Assumptions And An Analysis of DHIES. In David Naccache, editor, *Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 143–158, San Francisco, CA, USA, April 8–12, 2001. Springer-Verlag. 2.2

BMV08. Emmanuel Bresson, Jean Monnerat, and Damien Vergnaud. Separation Results on The "One-More" Computational Problems. In Tal Malkin, editor, *Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008*, volume 4964 of *Lecture Notes in Computer Science*, pages 71–87, San Francisco, CA, USA, April 8–11, 2008. Springer-Verlag. 1

Bro07. Daniel R. L. Brown. Irreducibility to The One-More Evaluation Problems: More May Be Less. Technical Report 2008/435, International Association for Cryptologic Research, 2007. Available at http://eprint.iacr.org/2007/435. 1, 2.1, 2.2

Che06. Jung Hee Cheon. Security Analysis of The Strong Diffie-Hellman Problem. In Serge Vaudenay, editor, *Advances in Cryptology — EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 1–11, St. Petersburg, Russia, May 28–June 1, 2006. Springer-Verlag. 1, 5

CKS08. David Cash, Eike Kiltz, and Victor Shoup. The Twin Diffie-Hellman Problem And Applications. In Nigel Smart, editor, *Advances in Cryptology — EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 127–145, Istanbul, Turkey, April 13–17, 2008. Springer-Verlag. 1, 3.1

Dam91. Ivan Damgård. Towards Practical Public Key Systems Secure against Chosen Ciphertext Attacks. In Joan Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 445–456, Santa Barbara, California, USA, August 11–15, 1991. Springer-Verlag, 1992. 1, 2.3

Elg85. Taher Elgamal. A Public Key Cryptosystem And A Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985. 1, 2.3

Gjø06. Kristian Gjøsteen. A New Security Proof for Damgård's ElGamal. In David Pointcheval, editor, *Topics in Cryptology - CT-RSA 2006, The Cryptographers' Track at the RSA Conference 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 150–158, San Jose, CA, USA, February 13–17, 2006. Springer-Verlag. 1, 2.2, 2.3

OP01. Tatsuaki Okamoto and David Pointcheval. The Gap-Problems: A New Class of Problems for The Security of Cryptographic Schemes. In Kwangjo Kim, editor, *Public Key Cryptography 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 104–118, Cheju Island, Korea, February 13–15, 2001. Springer-Verlag. 2.2, 5

Sho97. Victor Shoup. Lower Bounds for Discrete Logarithms And Related Problems. In Walter Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266, Konstanz, Germany, 11–15 May 1997. Springer-Verlag. 5

TY98. Yannis Tsiounis and Moti Yung. On The Security of ElGamal-Based Encryption. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography 1998*, volume 1431 of *Lecture Notes in Computer Science*, pages 117–134, Pacifico Yokohama, Japan, 5–6 February 1998. Springer-Verlag. 1, 2.3