# Chosen-Ciphertext Secure Key-Encapsulation Based on Gap Hashed Diffie-Hellman

Eike Kiltz

CWI Amsterdam
The Netherlands
kiltz@cwi.nl
http://kiltz.net

**Abstract.** We propose a practical key encapsulation mechanism with a simple and intuitive design concept. Security against chosen-ciphertext attacks can be proved in the standard model under a new assumption, the Gap Hashed Diffie-Hellman (GHDH) assumption. The security reduction is tight and simple.
Secure key encapsulation, combined with an appropriately secure symmetric encryption scheme, yields a hybrid public-key encryption scheme which is secure against chosen-ciphertext attacks. The implied encryption scheme is very efficient: compared to the previously most efficient scheme by Kurosawa and Desmedt [Crypto 2004] it has 128 bits shorter ciphertexts, between 25-50% shorter public/secret keys, and it is slightly more efficient in terms of encryption/decryption speed. Furthermore, our scheme enjoys (the option of) public verifiability of the ciphertexts and it inherits all practical advantages of secure hybrid encryption.

## 1 Introduction

One of the main fields of interest in cryptography is the design and the analysis of the security of encryption schemes in the public-key setting (PKE schemes). In this work our goal is to provide schemes for which we can provide theoretical proofs of security (without relying on heuristics such as the random oracle), but which are also efficient and practical.

Key Encapsulation. Instead of providing the full functionality of a public-key encryption scheme, in many applications it is sufficient to let sender and receiver agree on a common random session key. This can be accomplished with a *key encapsulation mechanism* (KEM) as formalized by Cramer and Shoup [11]. In this protocol a sender (knowing the receivers public key) runs an encapsulation algorithm to produce a random session key together with a corresponding ciphertext. This ciphertext is sent (over a potentially insecure channel) to the receiver, who (using his secret key) can uniquely reconstruct the session key using a decapsulation algorithm. In the end both parties share a common random session key. A strong notion of security (security against *chosen-ciphertext attacks* [23]) requires that, roughly, not even an active eavesdropper (interacting

with a decapsulation oracle that allows him to obtain session keys corresponding to ciphertexts of his choosing) can learn any information about the random session key corresponding to a given ciphertext. After the execution of the protocol the random session key may now be used for arbitrary symmetric-key operations such as a symmetric encryption scheme. If both, the KEM and the symmetric primitive, are secure against chosen-ciphertext attacks then composition theorems are used to obtain the same security guarantees for the hybrid encryption protocol.

In this work we are interested in designing key encapsulation mechanisms that are both efficient and provably secure with respect to a reasonable intractability assumption. To motivate our approach we start with some history on key encapsulation.

Diffie-Hellman Key Encapsulation. In the Diffie-Hellman key encapsulation mechanism [12] the receiver's public key consists of the group element $g^x$ (we assume a commutative cyclic group of prime order and generator $g$ to be given), the secret key of the random index $x$. Key encapsulation is done by computing the ciphertext as $g^y$ for random $y$; the corresponding session key is the group element $g^{xy} = (g^x)^y$ (and therefore called *Diffie-Hellman Key*). This key is recovered from the ciphertext by the possessor of the secret key $x$ by computing $g^{xy}$ as $(g^y)^x$. In practice one mostly requires the session key to be a binary string rather of fixed length than a group element. This is overcome by feeding the Diffie-Hellman key $g^{xy}$ to a hash function $\mathsf{H}$ with binary image to obtain a session key $\mathsf{H}(g^{xy})$. This simple key encapsulation scheme can be proved secure against chosen-plaintext attacks under the *Hashed Diffie-Hellman* (HDH) assumption, as formalized in [1]. The HDH assumption (relative to a hash function $\mathsf{H}$) states, roughly, that the two distributions $(g^x, g^y, \mathsf{H}(g^{xy}))$ and $(g^x, g^y, R)$ for random indices $x, y$ and a random bit-string $R$ (of appropriate length) are computational indistinguishable. Under the HDH assumption, Hashed Diffie-Hellman can be proven secure against chosen-plaintext attacks (IND-CPA).

For various reasons, the stronger notion of chosen-ciphertext (IND-CCA) security [23] has emerged as the "right" notion of security for key encapsulation and encryption. Hashed Diffie-Hellman will be our starting point and the goal will be to modify the scheme in order to obtain security against chosen-ciphertext attacks under a reasonable intractability assumption.

Our Construction. We modify the Hashed Diffie-Hellman key encapsulation in order to obtain a KEM that is provably secure against chosen-ciphertext attacks under the *Gap Hashed Diffie-Hellman assumption* (to be introduced later). Our main idea is to add some redundant information to the ciphertext of the Hashed Diffie-Hellman key encapsulation. This information is used to check if a given ciphertext was properly generated by the encapsulation algorithm (and hence is "consistent"); if the ciphertext is consistent then decapsulation returns the session key, otherwise it simply rejects. Our scheme's security relies on the Gap Hashed Diffie-Hellman (GHDH) assumption which states that, roughly, the two distributions $(g^x, g^y, \mathsf{H}(g^{xy}))$ and $(g^x, g^y, R)$ are hard to distinguish even relative to a "Diffie-Hellman oracle" that efficiently distinguishes $(g^x, g^y, g^{xy})$ from

$(g^x, g^y, g^z)$. Here the term "gap" stems from the fact that there is a gap between the Decisional and the Computational version of the Diffie-Hellman problem: the computational problem is hard to solve even though the corresponding decisional problem is easy.

Main Results. Our main result shows that our key encapsulation mechanism is secure against chosen-ciphertext attacks assuming the GHDH assumption holds. The scheme has very short ciphertexts (2 groups elements or approximately 512 bits for 128 bits security) and its security reduction is *tight*. When our scheme gets instantiated in gap-groups [20] a given ciphertext can get checked for consistency solely based on the knowledge of the public key. This feature (sometimes called "public verifiability of the ciphertext") has proved very useful, e.g. for building a chosen-ciphertext secure threshold encapsulation scheme [9]. Furthermore, we show that our framework extends to building KEMs based on the Gap Hashed Multi Diffie-Hellman (GHMDH) assumption, a natural generalization of GHDH with potentially stronger security properties. The GHMDH assumption states that given many independent Diffie-Hellman instances $(g_i, h_i, g_i^{r_i})_{1 \le i \le \ell}$, evaluating the $\ell^2$ possible (hidden) Diffie-Hellman keys $(h_i)^{r_j}$ $(1 \le i, j \le \ell)$ on a fixed public predicate $\mathsf{H} : \mathbb{G}^{\ell \times \ell} \to \mathbb{G}$ yields an element that is indistinguishable from a random one, even relative to a DDH oracle. The GHMDH assumption in particular includes (a paring-free variant of) the Gap Linear Diffie-Hellman (GLDH) assumption [6].

Related Work. Cramer and Shoup [10, 11] proposed the first practical public-key encryption scheme in the standard model. More recently, Kurosawa and Desmedt came up with a direct hybrid encryption scheme [19] improving the performance of the original CS scheme both in computational efficiency and in ciphertext length. In their hybrid construction the symmetric scheme has to be secure in the sense of *authenticated encryption* [2] which is a strictly stronger security requirement than in the standard KEM/DEM hybrid paradigm [11], and in particular it necessarily adds 128 bits of redundancy to the symmetric ciphertext. The KD-KEM (i.e. the KEM part of the Kurosawa Desmedt hybrid encryption scheme) is similar to our KEM construction. In fact, the KD-KEM can be obtained from our KEM by (roughly) switching the symmetric key with one element from the ciphertext. Our scheme can be proved chosen-ciphertext secure whereas there exists a simple chosen-ciphertext attack against the KD-KEM [14]. We think that this is really a surprising fact since a small difference in the constellation of the ciphertexts seems to turn the scale when it comes to security of the two schemes.

An alternative group of schemes ("IBE-based schemes") is based on recent results [7, 16] observing that identity-based encryption (IBE) implies chosen-ciphertext secure encryption. The recent approach taken by Boyen, Mei, and Waters [9] was to improve efficiency of one particular instantiation [5] (based on the BDH assumption) obtained by the above IBE transformation. Similar results were also obtained independently by Kiltz [16]. All the encryption schemes constructed this way, however, so far remained less efficient than the reference scheme from Kurosawa-Desmedt. Our KEM constructions based on GHDH and

GHMDH are related (and generalize) the KEMs obtained in [9, 16] and therefore fits best into the latter class of IBE-based [7, 16] schemes (even though they are not derived from any IBE scheme).

Discussion and Comparison. Our porposed hybrid PKE scheme based on GHDH is more efficient than the "reference scheme" by Kurosawa and Desmedt [19]: it has "one MAC" shorter ciphertexts (by combining it with redundancy-free symmetric encryption [21]), between 25-50% shorter public/secret keys, and it is slightly more efficient in terms of encryption/decryption. However, an arguable disadvantage of our scheme is that security can only be proven on the new GHDH assumption, whereas security of the KD scheme provably relies on the well-established and purely algebraic DDH assumption. An extensive comparison with all known KEM/PKE schemes in the standard model is done in Table 1 (Section 5).

Recent Results. Recently, building on this work, Hofheinz and Kiltz [15] combined a variation of our scheme with symmetric authenticated encryption (and hence adding 128 bits redundancy to the ciphertexts) to obtain public-key encryption secure under the DDH assumption. Their technique also extends to the more general class of (Hashed) Multi Diffie-Hellman assumptions which can be seen as the "DDH-oracle free" variant of HGMDH.

Full Version. A full version of this extended abstract is available on the Cryptology ePrint archive [18].

## 2 Public Key Encapsulation Mechanisms

A *public-key encapsulation* (KEM) scheme $\mathcal{KEM} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ with key-space $\mathsf{KeySp}(k)$ consists of three polynomial-time algorithms. Via $(pk, sk) \xleftarrow{\$} \mathsf{Kg}(1^k)$ the randomized key-generation algorithm produces keys for security parameter $k \in \mathbb{N}$; via $(K, C) \xleftarrow{\$} \mathsf{Enc}(1^k, pk)$ a key $K \in \mathsf{KeySp}(k)$ together with a ciphertext $C$ is created; via $K \leftarrow \mathsf{Dec}(sk, C)$ the possessor of secret key $sk$ decrypts ciphertext $C$ to get back a key. For consistency, we require that for all $k \in \mathbb{N}$, and all $(K, C) \xleftarrow{\$} \mathsf{Enc}(1^k, pk)$ we have $\Pr[\,\mathsf{Dec}(C) = K\,] = 1$, where the probability is taken over the choice of $(pk, sk) \xleftarrow{\$} \mathsf{Kg}(1^k)$, and the coins of all the algorithms in the expression above.

We require the KEM to be secure against chosen-ciphertext attacks. Formally, we associate to an adversary $\mathcal{A}$ the following experiment:

**Experiment** $\mathbf{Exp}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)$
$(pk, sk) \xleftarrow{\$} \mathsf{Kg}(1^k)$ ; $K_0^* \xleftarrow{\$} \mathsf{KeySp}(k)$ ; $(K_1^*, C^*) \xleftarrow{\$} \mathsf{Enc}(pk)$
$\delta \xleftarrow{\$} \{0,1\}$ ; $\delta' \xleftarrow{\$} \mathcal{A}^{\mathsf{DecO}(sk,\cdot)}(pk, K_\delta^*, C^*)$
If $\delta \neq \delta'$ then return 0 else return 1

where the oracle $\mathsf{DecO}(sk, \cdot)$ queried on $C$ returns $K \leftarrow \mathsf{Dec}(sk, C)$ with the restriction that $\mathcal{A}$ is not allowed to query $\mathsf{DecO}(sk, \cdot)$ on the target ciphertext

$C^*$. We define the advantage of $\mathcal{A}$ in the left experiment as

$$\mathbf{Adv}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k) \;=\; \left|\Pr\left[\,\mathbf{Exp}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)=1\,\right]-\frac{1}{2}\right|.$$

A key encapsulation mechanism $\mathcal{KEM}$ is said to be *indistinguishable against chosen-ciphertext attacks* (IND-CCA) if the advantage function $\mathbf{Adv}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)$ is a negligible function in $k$ for all polynomial-time adversaries $\mathcal{A}$.

Note that in contrast to the original definition given by Cramer and Shoup [11] we consider a simplified (but equivalent) security experiment without a "find-stage".

## 3 Complexity Assumptions

### 3.1 Standard Diffie-Hellman assumptions

We first start with the following well known standard assumptions which we review for completeness. The *Computational Diffie-Hellman assumption* (CDH) states, that given the input $(g, g^x, g^y)$ where $x, y$ are drawn at random from $\mathbb{Z}_p$ ($g$ is a generator of a group $\mathbb{G}$ of prime order $p$), it should be computationally infeasible to compute $g^{xy}$. However, under the CDH assumption it might be as well possible to efficiently compute some information about $g^{xy}$, say a single bit of the binary representation or even all but super-logarithmically many bits. A stronger assumption that has been gaining popularity is the *Decisional Diffie-Hellman assumption* (DDH). It states, roughly, that the distributions $(g, g^x, g^y, g^{xy})$ and $(g, g^x, g^y, g^z)$ are computationally indistinguishable when $x, y, z$ are drawn at random from $\mathbb{Z}_p$. Another variant of the Diffie-Hellman assumption is the *Gap Diffie-Hellman assumption* (GDH). It states that the CDH assumption is still hard even though an adversary has additional access to an oracle that solves the DDH problem.

### 3.2 The Gap Hashed Diffie-Hellman assumption

As indicated above, semantic security requires that we will be able to get some number of hard-core bits from the Diffie-Hellman key (i.e. bits that cannot be distinguished from random bits). We will be using a gap-assumption relative to a DDH oracle, so clearly we are not allowed to take the whole Diffie-Hellman key. Our assumption is that applying a suitable hash function $\mathsf{H}$ (for example, a cryptographic hash function like SHA-1) to $g^{xy}$ will yield such bits. The assumption we make, called the Gap Hashed Diffie-Hellman assumption (GHDH) is a "composite one"; it concerns the interaction between a hash function $\mathsf{H}$ and the group $\mathbb{G}$. The GHDH is an extension of the HDH assumption formalized by Abdalla, Bellare, Rogaway [1].

Our schemes will be parameterized by a *parameter generator*. This is a polynomial-time algorithm $\mathsf{Gen}$ that on input $1^k$ returns the description of a multiplicative cyclic group $\mathbb{G}$ of prime order $p$, where $2^k < p < 2^{k+1}$, and a

random generator $g$ of $\mathbb{G}$. $\mathsf{Gen}$ furthermore outputs the description of a random hash function $\mathsf{H} : \mathbb{G} \to \{0,1\}^{l(k)}$ that outputs $l(k)$ bits for a fixed polynomial $l(\cdot)$. Throughout the paper we use $\mathcal{HG} = (\mathbb{G}, g, p, \mathsf{H})$ as shorthand for the description of the hash group obtained by running $\mathsf{Gen}$.

The GHDH assumption relative to $\mathsf{Gen}$ states that the two distributions $(g^x, g^y, \mathsf{H}(g^{xy}))$ and $(g^x, g^y, R)$ are computationally indistinguishable when $x, y$ are drawn at random from $\mathbb{Z}_p$ and $R$ is drawn at random from $\{0,1\}^{l(k)}$. This assumption should hold relative to an oracle that efficiently solves the DDH problem. More formally, to an adversary $\mathcal{B}$ we associate the following experiment.

**Experiment** $\mathbf{Exp}^{\mathrm{ghdh}}_{\mathsf{Gen},\mathsf{H},\mathcal{B}}(1^k)$
$\mathcal{HG} \xleftarrow{\$} \mathsf{Gen}(1^k)$ ; $x, y \xleftarrow{\$} \mathbb{Z}_p^*$ ; $W_0 \xleftarrow{\$} \{0,1\}^{l(k)}$ ; $W_1 \leftarrow \mathsf{H}(g^{xy})$
$\gamma \xleftarrow{\$} \{0,1\}$ ; $\gamma' \xleftarrow{\$} \mathcal{B}^{\mathsf{DDHsolve}_{\mathbb{G}}(\cdot,\cdot,\cdot,\cdot)}(1^k, \mathcal{HG}, g^x, g^y, W_\gamma)$
If $\gamma \neq \gamma'$ then return 0 else return 1

Here the oracle $\mathsf{DDHsolve}_{\mathbb{G}}(g, g^a, g^b, g^c)$ returns 1 iff $ab = c \bmod p$. We define the advantage of $\mathcal{B}$ in the above experiment as

$$\mathbf{Adv}^{\mathrm{ghdh}}_{\mathsf{Gen},\mathcal{B}}(k) \;=\; \left| \Pr\left[ \mathbf{Exp}^{\mathrm{ghdh}}_{\mathsf{Gen},\mathcal{B}}(1^k) = 1 \right] - \frac{1}{2} \right|.$$

We say that the *Gap Hashed Diffie-Hellman (GHDH) assumption relative to group generator* $\mathsf{Gen}$ holds if $\mathbf{Adv}^{\mathrm{ghdh}}_{\mathsf{Gen},\mathcal{B}}$ is a negligible function in $k$ for all polynomial-time adversaries $\mathcal{B}$.

We remark that in so called gap-groups, i.e. in groups where the Decisional Diffie-Hellman (DDH) problem is easy on every input while the computational Diffie-Hellman (CDH) problem CDH problem is hard [20], the GHDH assumption is equivalent to the HDH assumption. A possible implementation of gap-groups is given by the Weil/Tate bilinear pairing allowing to efficiently compute a bilinear pairing which can be used to solve DDH [8].

At first glance one may argue that assuming the hashed key $\mathsf{H}(g^{xy})$ to be indistinguishable from a random string even though we can efficiently distinguish $g^{xy}$ from a random group element sounds quite unreasonable and that, in a sense, hardness falls back on "random-oracle-like" properties of the hash function. However, this intuition is not true. We can show that in generic groups [24] GHDH holds (unconditionally) assuming the hash function $\mathsf{H}$ is "weakly one-way". The latter result basically means that the GHDH assumption depends on the hardness of computing the Diffie-Hellman key plus the fact that given only $\mathsf{H}(g^{xy})$ it is hard to recover sufficient information on the Diffie-Hellman key $g^{xy}$. This should in particular hold for cryptographic hash functions like SHA-1. Also, the well known and often employed Bilinear Diffie-Hellman (BDH) assumption [8] can in fact be seen as a special (algebraic) instantiation of the GHDH assumption. More precisely, using the specific algebraic hash function $\mathsf{H}(X) := \hat{e}_Z(X)$, where $\hat{e}_Z(X) := \hat{e}(X, Z)$ is a bilinear mapping for fixed $Z = g^z$ (but chosen uniformly at setup), we get $\mathsf{H}(g^{xy}) = \hat{e}(g^{xy}, g^z) = \hat{e}(g, g)^{xyz}$ and GHDH actually gets BDH (here the output of $\mathsf{H}$ is a group element, not a binary string). In this

context, GHDH instantiated with a cryptographic hash function appears not to be a less reasonable assumption than the "standard" BDH assumption.

More details are given in the full version [18]. There we also propose various candidates for $\mathsf{Gen}$ (i.e., for the prime-order group $\mathbb{G}$ and the hash function $\mathsf{H}$) and provide a detailed security analysis of the GHDH assumption. In practice however, we recommend using a cryptographic hash function like MD5 or SHA-1.

## 4 Key Encapsulation based on GHDH

### 4.1 The Key Encapsulation Mechanism

Let $\mathcal{HG} = (\mathbb{G}, g, p, \mathsf{H})$ be random parameters obtained by running the parameter algorithm $\mathsf{Gen}(1^k)$, where $\mathsf{H} : \mathbb{G} \to \{0,1\}^{l(k)}$ is a random instance of a hash function such that the GHDH assumptions holds relative to $\mathsf{Gen}$. Let $\mathsf{INJ} : \mathbb{G} \to \mathbb{Z}_p$ be an efficiently computable injective encoding.[1] We build a key encapsulation mechanism $\mathcal{KEM} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ as follows.

| $\mathsf{Kg}(1^k)$ | $\mathsf{Enc}(pk)$ | $\mathsf{Dec}(sk, C)$ |
|---|---|---|
| $x, y \xleftarrow{\$} \mathbb{Z}_p^*$ | $r \xleftarrow{\$} \mathbb{Z}_p^*$ ; $c \leftarrow g^r$ | Parse $C$ as $(c, \pi)$ |
| $u \leftarrow g^x$ ; $v \leftarrow g^y$ | $t \leftarrow \mathsf{INJ}(c)$ ; $\pi \leftarrow (u^t v)^r$ | $t \leftarrow \mathsf{INJ}(c)$ |
| $pk \leftarrow (u, v)$ | $K \leftarrow \mathsf{H}(u^r) \in \{0,1\}^{l(k)}$ | If $c^{xt+y} \neq \pi$ then $\perp$ |
| $sk \leftarrow (x, y)$ | $C \leftarrow (c, \pi) \in \mathbb{G}^2$ | Else $K \leftarrow \mathsf{H}(c^x)$ |
| Return $(pk, sk)$ | Return $(C, K)$ | Return $K$ |

Decapsulation also has to perform one subgroup-membership test, i.e. it checks if $c \in \mathbb{G}$, and returns $\perp$ (reject) otherwise. Note that $c^{xt+y} = \pi$ then automatically also implies $\pi \in \mathbb{G}$.

Efficiency. The public key contains two group elements, the secret key of two elements from $\mathbb{Z}_p$. A ciphertext $C$ consists of two group elements, the key $K$ is a binary string of length $l(k)$. Ignoring all "symmetric operations", encapsulation needs three regular exponentiations, whereas decapsulation can be carried out in two exponentiation. Using the concept sequential/multi-exponentiations[2] (see, e.g., [22, 4]) a considerable (and practical) speed-up can be obtained: encapsulation needs two regular exponentiations (to compute $c$ and $K$) plus one multi-exponentiation (to compute $\pi = u^{tr} v^r$), whereas decapsulation can be carried out in one single sequential exponentiation (to compute $c^{xt+y}$ and $c^x$).

[1] Actually, an "almost injective" encoding is sufficient for our purpose, see [9]. Most interesting groups allow for such an encoding [11, 9, 13]. If such an encoding is not available one can also use a target collission resistant hash function $\mathsf{TCR} : \mathbb{G} \to \mathbb{Z}_p$, see [18] for more details.

[2] One multi-exponentiaion computes the group element $g^a h^b$ and one sequential exponentiation computes the two group elements $g^a$ and $g^b$ in one single step (for the same fixed base $g$). Both concepts are related and (using Pippenger's algorithm [22]) can be carried out in about $(1 + 2/\log\log p) \log p$ multiplications over $\mathbb{G}$ [4] which we will count as $\approx 1.2$ exponentiations.

CORRECTNESS. Fix a pair of keys $(pk, sk)$. We call a ciphertext $C = (c, \pi) \in \mathbb{G}^2$ *consistent* if $c^{xt+y} = \pi$ for $t = \mathsf{INJ}(c)$. For a correctly generated ciphertext $C = (c, \pi) = (g^r, u^{tr}v^r)$ we have $c^{xt+y} = (g^{xt+y})^r = (u^t v)^r = \pi$ and hence $C$ is consistent. In that case decapsulation reconstructs the session key as $K = \mathsf{H}(c^x) = \mathsf{H}((g^r)^x) = \mathsf{H}(u^r)$, as the key in encapsulation.

PUBLIC VERIFIABILITY IN GAP-GROUPS. Let $C = (c, \pi) \in \mathbb{G}^2$ be a ciphertext with $c = g^r$ for some value $r \in \mathbb{Z}_p$. Then $(g, u^t v = g^{xt+y}, c = g^r, \pi)$ is a Diffie-Hellman-tuple if and only if $g^{(xt+y)\cdot r} = \pi$ what is equivalent to $c^{xt+y} = \pi$. Therefore in gap-groups consistency of a ciphertext can be publicly checked using one call to the Diffie-Hellman oracle, i.e. by verifying if $\mathsf{DDHsolve}_{\mathbb{G}}(g, u^t v = g^{xt+y}, c = g^r, \pi)$ returns true. This property is denoted as *public verifiability of the ciphertext* and it give rise to a public-key threshold KEM [9].

### 4.2 Security

Our main theorem can be stated as follows:

**Theorem 1.** *Under the Gap Hashed Diffie-Hellman assumption relative to generator* $\mathsf{Gen}$*, the key encapsulation mechanism from Section 4.1 is secure against chosen-ciphertext attacks. In particular, for any adversary* $\mathcal{A}$ *against the KEM running for time* $\mathbf{Time}_{\mathcal{A}}(k)$*, there exist an adversary* $\mathcal{B}$ *with* $\mathbf{Adv}^{\mathrm{ghdh}}_{\mathsf{Gen},\mathcal{B}}(k) \geq \mathbf{Adv}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)$ *and* $\mathbf{Time}_{\mathcal{B}}(k) = \mathbf{Time}_{\mathcal{A}}(k) + \mathcal{O}(q \cdot \mathbf{Time}_{\mathbb{G}}(k))$*, where* $q$ *is an upper bound on the number of decapsulation queries made by adversary* $\mathcal{A}$ *and* $\mathbf{Time}_{\mathbb{G}}(k)$ *is the time for a standard operation in* $\mathbb{G}$*.*

We want to stress that the key encapsulation mechanism does not make use of the Decision Diffie-Hellman oracle $\mathsf{DDHsolve}$. Its existence is part of the assumption and solely needed for the proof of security.

The proof is quite simple. An intuitive way to understand it is as follows: first consider a modified KEM that is obtained by abandoning the hash function $\mathsf{H}$ from the construction in Section 4.1, i.e. the symmetric key is now computed as $K = u^r$. What we can prove is that this modified KEM is *one-way chosen-ciphertext secure* under the gap Diffie-Hellman (GDH) assumption. In the security reduction the DDH oracle provided by the GDH assumption is used to reject (as in the original scheme) every invalid ciphertext submitted by the adversary to the decryption oracle. The key idea of the reduction is based on an algebraic technique from [5] that was also used in [9, 16] in the context of KEMs. An attacker $\mathcal{B}$ against the GDH problem can setup the public-key for the adversary attacking the security of the KEM in a way that (i) adversary $\mathcal{B}$ (without knowing the secret key) can decapsulate every ciphertexts except the challenge ciphertext; (ii) decapsulating the challenge ciphertext is equivalent to solving GDH. If the adversary against the KEM is successfull (i.e. it decapsulates the challenge ciphertext) so this adversary can be used to break the GDH problem using the above simulation.

More details. Adversary $\mathcal{B}$ inputs a GDH instance $(g, u, g^a)$ and it's goal is to compute $T = u^a$ (recall that we are attacking one-way chosen-ciphertext security). He picks a random value $d$ and defines the (thereby correctly distributed)

public key as $pk = (u, v = u^{-t^*} g^d)$, where $t^* = \mathsf{INJ}(g^a)$. Note that this way a consistent ciphertext $(c, \pi)$ properly created by the encapsulation algorithm has the form

$$c = g^r, \quad \pi = (u^t v)^r = (u^r)^{t-t^*} c^d \,, \tag{1}$$

for some $t \in \mathbb{Z}_p$. Hence, in order to decapsulate the challenge ciphertext $C^* = (c^*, \pi^*)$ defined as $c^* := g^a$, $\pi^* := (g^a)^d = (u^a)^{t^*-t^*} c^d$ (i.e., a ciphertext computed with unknown randomness $a$ from the GDH instance, where $t^* = \mathsf{INJ}(c^*)$), adversary $\mathcal{A}$ (which is run on $pk$ and $C^*$) has to compute the target key $K^* = u^a$ what is equivalent to breaking GDH. On the other hand, for a decapsulation query for ciphertext $(c, \pi)$, $\mathcal{B}$ first checks for consistency using the DDH oracle $\mathsf{DDHsolve}$ provided by the GDH assumption. If the ciphertext is inconsistent it gets rejected. Otherwise, by injectivity of $\mathsf{INJ}$, we have $t = \mathsf{INJ}(c) \neq \mathsf{INJ}(c^*) = t^*$ and the correct key $K = u^r$ can be reconstructed by Eqn. (1) as $K = (\pi / c^d)^{1/(t-t^*)}$.

The step to full security (i.e., indistinguishability compared to one-wayness) now can be intuitively understood by the fact that (in terms of the assumption) we move from GDH to GHDH, i.e. under GHDH the hash function $\mathsf{H}$ hides all information about the Diffie-Hellman key $u^r$. A more formal proof is given in Appendix A.

### 4.3 KEM/DEM: from KEM to full encryption

A KEM and a symmetric encryption scheme (aka DEM) can be used to obtain a hybrid public-key encryption scheme. It is well known that if both the KEM and the DEM are chosen-ciphertext secure, then the resulting hybrid encryption is also chosen-ciphertext secure [11, Sec. 7]. The security reduction is tight. A DEM secure against chosen-ciphertext attacks can be built from relatively weak primitives, i.e. from any one-time symmetric encryption scheme by essentially adding a MAC. Phan and Pointcheval [21] showed that strong pseudorandomn permutations directly imply redundancy-free chosen-ciphertext secure DEMs that avoid the usual overhead due to the MAC. It seems reasonable to believe that known block-ciphers (auch as AES) are strong PRPs.

In the full version [18] we also sketch how to obtain a direct PKE scheme that may be usefull to non-interactive chosen-ciphertext secure threshold encryption scheme [9].

### 4.4 Relation to other encryption schemes

The KEMs based on "identity-based techniques" [9, 16, 17] are very similar to our construction. In fact, (a slight variation of) the KEM from [9] (which itself is based on the first IBE scheme Boneh and Boyen [5]) can be obtained from our KEM by instantiating the hash function $\mathsf{H}$ with a bilinear map, i.e. by defining $\mathsf{H}(X) = \hat{e}(g^z, X)$ (further simplifications in the decapsulation algorithm must be applied). As we already sketched in Section 3.2, security of the KEM then can be proved relative to the BDH assumption (just as in [9]). However, since it

involves computing bilinear maps, the BWM-KEM is considerably less efficient than our proposal when $\mathsf{H}$ is a cryptographic hash function.

Surprisingly, the KEM part (KD-KEM) of the Kurosawa-Desmedt public-key encryption scheme [19] looks quite similar to our construction. Indeed, the KD-KEM encapsulates by computing the ciphertext as $(c_1, c_2) = (g^r, \hat{g}^r)$ and the corresponding symmetric key is defined as $K = (u^t v)^r$, where $g, \hat{g}, u = g^x \hat{g}^{\hat{x}}, v = g^y \hat{g}^{\hat{y}}$ are elements from the public key and $t$ is computed as $t = \mathsf{TCR}(c_1, c_2)$. In comparison (and ignoring the hash function) our scheme basically *swaps* the elements $c_2$ and $K$, i.e. the ciphertexts of our scheme are given by $(g^r, (u^t v)^r)$ ($t$ now only depends on $g^r$), where the corresponding key is $\mathsf{H}(u^r)$.

In contrast our KEM is provably secure under a well-defined number-theoretic assumption whereas the KD-KEM was recently shown to be not even one-way chosen-ciphertext secure [14]. One could possible remark that the stronger security properties of our KEM inherently rely on the stronger assumption, i.e., the hash function $\mathsf{H}$ and the DDH oracle in the GHDH assumption (the gap-property). However, this is not true as we will explain now; security rather seems to depend on the particular constellation of the ciphertexts of our KEM. First, the attack from [14] aganst the KD-KEM is still valid if the two elements in the KD-KEM ciphertext get checked for consistency before decapsulating the key, i.e. the attack does not rely on "inconsistent ciphertext queries". In other words it is not the "gap"-property of the GHDH assumptions that makes the difference in the (in-)security of the two KEMs. Second, chosen-ciphertext security of our KEM does also not depend on the hash function $\mathsf{H}$ since without $\mathsf{H}$ our KEM is still *one-way* chosen-ciphertext secure under the gap computational Diffie-Hellman assumption. As pointed out earlier the hash function $\mathsf{H}$ is only responsible to provide indistinguishability (rather than one-wayness).

### 4.5 Key-encapsulation based on GHMDH

In this section we sketch a usefull generalization of our KEM construction to build schemes based on the general class of GHMDH assumptions which we now introduce.

For an integer $\ell \geq 1$ let $\mathbf{D} \in \mathbb{G}^{\ell \times \ell}$ be a matrix with entries $D_{i,j} \in \mathbb{G}$ ($1 \leq i, j \leq \ell$). Let $\mathsf{H} : \mathbb{G}^{\ell \times \ell} \to \mathcal{K}$ be a hash-function that maps $\ell^2$ group elements into a key-space $\mathcal{K}$. Informally, the Gap Hashed Multi Diffie-Hellman (GHMDH) assumption (realtive to hash function $\mathsf{H}$ and group $\mathbb{G}$) states that, given $g_1, \ldots, g_\ell, h_1, \ldots, h_\ell, g_1^{r_1}, \ldots, g_\ell^{r_\ell}$ and access to a DDH oracle, it is computationally infeasible to distinguish $\mathsf{H}(\mathbf{D})$ from a random element in $\mathcal{K}$, where the (hidden) entries of matrix $\mathbf{D}$ contain all $\ell^2$ possible combinations of Diffie-Hellman keys, i.e. $D_{i,j} = h_i^{r_j}$. Intuitively, the hash function $\mathsf{H}$ can be viewed as a hard predicate of the $\ell^2$ different Diffie-Hellman keys. Clearly, for $\ell = 1$ and $\mathcal{K} = \{0,1\}^{l(k)}$ this simplifies to the GHDH assumption but in this section we focus mostly on algebraic candidates of the form $\mathsf{H} : \mathbb{G}^{\ell \times \ell} \to \mathbb{G}$, for $\ell \geq 2$.

As one illustrating example of the much general class of GHMDH assumptions, the Gap Decision Linear Diffie-Hellman (GLDH) assumption [6] is obtained by setting $\ell = 2$ and defining $\mathsf{H} : \mathbb{G}^{2\times 2} \to \mathbb{G}$ as $\mathsf{H}(\mathbf{D}) = D_{1,1} \cdot D_{1,2}$. More

precisely, the GLDH assumption states that, given $g_1, g_2, g_1^{r_1}, g_2^{r_2}, h_1, W$, destinguishing $W = h_1^{r_1+r_2}$ from a uniform $W \in \mathbb{G}$ is computational infeasible, even relative to a DDH oracle. Originally, the GLDH assumption was defined over bilinear maps [6] (called Decision Linear Diffie-Hellman assumption), whereas here we only require the assumption to hold relative to a DDH oracle. This, in particilar, makes it possible to define (and apply) it relative to any cyclic group [16].

More generally, for any polynomial $\ell = \ell(k) \geq 2$, one can also define the class of $\ell$-GLDH assumptions for arbitrary $\ell = \ell(k) = \text{poly}(k)$ by defining $\mathsf{H} : \mathbb{G}^{\ell \times \ell} \to \mathbb{G}$ as $\mathsf{H}(\mathbf{D}) = \prod_{i=1}^{\ell} D_{1,i}$. (Note that the 1-GLDH assumption states that DDH is hard relative to a DDH oracle which is clearly insecure without applying any further hash function to the Diffie-Hellman key.) The $\ell$-GLDH assumptions form a strict hierarchy of security assumptions with 2-GLDH = GLDH and, the larger the $\ell$, the weaker the $\ell$-GLDH assumption. More precisely, for any $\ell \geq 2$ we have that $\ell$-GLDH implies $\ell+1$-GLDH. On the other hand (extending [6]) we can show that in the generic group model [24], the $\ell+1$-GLDH assumption holds, even relative to an $\ell$-GLDH oracle.

We now (extending Section 4.1) build a key encapsulation mechanism $\mathcal{KEM} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ based on the HGMDH assumption. We define $S_{\mathsf{H}}$ as the subset of indices $(i,j) \in \{1,\ldots,\ell\}^2$ such that the hash function $\mathsf{H}(\mathbf{D})$ depends on entry $D_{i,j}$. (For example, for $\ell$-GLDH we have $S_{\mathsf{H}} = \{(1,1),\ldots,(1,\ell)\}$.) Let $\mathsf{TCR} : \mathbb{G}^{\ell} \to \mathbb{Z}_p$ be a target collision-resistant hash function.

Key generation $\mathsf{Kg}(1^k)$ generates random group elements $g_1,\ldots,g_\ell,h_1,\ldots,h_\ell$ and indices $x_{i,j}$ $((i,j) \in S_{\mathsf{H}})$ such that $h_i = g_j^{x_{i,j}}$. Furthermore it defines $u_{i,j} = g_j^{y_{i,j}}$, for random $y_{i,j}$ $((i,j) \in S_{\mathsf{H}})$. The public key contains the elements $(g_i)_{1\leq i\leq \ell}, (h_i)_{1\leq i\leq \ell}$, and $(u_{i,j})_{(i,j)\in S_{\mathsf{H}}}$, and the secret key contains all corresponding indices.

| $\mathsf{Enc}(pk)$ | $\mathsf{Dec}(sk, C)$ |
|---|---|
| For $j \in \{1,\ldots,\ell\}$: $r_j \xleftarrow{\$} \mathbb{Z}_p^*$ ; $c_j \leftarrow g_j^{r_j}$ | $t \leftarrow \mathsf{TCR}(c_1,\ldots,c_\ell)$ |
| $t \leftarrow \mathsf{TCR}(c_1,\ldots,c_\ell)$ | For each $(i,j) \in S_{\mathsf{H}}$: |
| For $(i,j) \in S_{\mathsf{H}}$: | if $c_j^{x_{i,j}t+y_{i,j}} \neq \pi_{i,j}$ $\perp$ |
| $\pi_{i,j} \leftarrow (h_i^t u_{i,j})^{r_j}$ ; $K_{i,j} \leftarrow h_i^{r_j}$ | $K_{i,j} \leftarrow c_j^{x_{i,j}}$ |
| $K \leftarrow \mathsf{H}(\mathbf{K})$ ; $C \leftarrow (c_1,\ldots,c_\ell,(\pi_{i,j})_{(i,j)\in S_{\mathsf{H}}})$ | Return $K \leftarrow \mathsf{H}(\mathbf{K})$ |
| Return $(C,K)$ | |

The ciphertexts of this KEM contain $\ell+|S_{\mathsf{H}}|$ group elements, public/secret keys $2\ell+|S_{\mathsf{H}}|$ elements. The above scheme instantiated with the 2-GLDH assumption reproduces the KEM from [16] which, for any polynomial $\ell \geq 2$, generalizes to the class of $\ell$-GLDH schemes. Using simiar techniques as for the proof of Theorem 1, the above scheme can be proved secure under the GHMDH assumption, see [18] for details.

## 5 Comparison

The usual efficiency comparison with all previously known chosen-ciphertext secure KEMs/encryption schemes in the standard model is assembled in Table 1. Here KD is the hybrid encryption scheme from Kurosawa and Desmedt [19] and

| Scheme | Security Assmptn | Cipher Overhead | Enc #pair+#[mult,reg]-exp | Dec #pair+#[mult,reg]-exp | Keysize (pk/sk) | Publ Vfy? | Any group? |
|---|---|---|---|---|---|---|---|
| KD | DDH | $2\|p\|$ 640 | $0+[1,2]$ | $0+[1,0]$ | 4/4 | — | $\surd$ |
| CS | DDH | $3\|p\|$ 768 | $0+[1,3]$ | $0+[1,1]$ | 5/5 | — | $\surd$ |
| BMW | BDH | $2\|p\|$ 512 | $0+[1,2]$ | $1+[0,1]$ | 4/3 | $\surd$ | — |
| **Ours §4.1** | GHDH | $2\|p\|$ 512 | $0+[1,2]$ | $0+[1,0]$ | 3/2 | $\surd^*$ | $\surd$ |
| **Ours §4.5** | $\ell$-GLDH | $2\ell\|p\|$ $512\ell$ | $0+[\ell,2\ell]$ | $0+[\ell,0]$ | $2\ell+1/2\ell$ | $\surd^*$ | $\surd$ |

*in gap and pairing groups only

**Table 1.** Efficiency comparison for chosen-ciphertext secure hybrid encryption schemes. Some figures are borrowed from [9, 16]. For efficiency we count the number of pairings + [multi exponentiations, regular exponentiations] used for encryption and decryption. All "symmetric" operations are ignored. Ciphertext overhead represents the difference (in bits) between ciphertext and plaintext length. For concreteness the expected ciphertext overhead for an 128-bit implementation is also given. The keysize is measured in two parameters: the size of the system parameters (which are fixed for every public-key) plus the size of the public key $pk$, and the size of the secret key $sk$. Here we only take into account the number of group elements for params plus $pk$, and the number of elements in $\mathbb{Z}_p^*$ for $sk$. A "$\surd$" in the "Publ. Vfy" column means that the scheme supports public verifiability. A "$\surd$" in the "Any group?" column means that the scheme can be implemented in any prime-order group, whereas a "—" means that the scheme has to be implemented in pairing groups. For comparison we mention that relative timings for the various operations are as follows: bilinear pairing $\approx 3-5$, multi(=sequential)-exponentiation $\approx 1.2$ [4], and regular exponentiation $= 1$.

---

CS refers to the Cramer-Shoup encryption scheme [10] which we compare in its hybrid variant from [11]. BMW is the KEM from Boyen, Mei, and Waters [9]. Our first scheme is the GHDH-based KEM from Section 4.1 instantiated with an efficient cryptographic hash function $\mathsf{H} : \mathbb{G} \to \{0,1\}^{l(k)}$. Our second scheme refers to the $\ell$-GLDH-based scheme from Section 4.5 which, for the case $\ell = 2$, simplifies to the GLDH-based KEM from [16]. All KEMs are assumed to be instatiated using a redundancy-free chosen-ciphertext secure symmetric scheme to obtain a full hybrid PKE scheme. The KD encryption scheme can only be proved secure in combination with an authenticated symmetric encryption scheme [2] which inherently adds "one MAC" overhead to the ciphertext size.

Even though our scheme shares the same number of exponentiations for encryption/decryption with the KD scheme, it has some practical advantages which makes a more efficient implementation possible. First, it is possible to use a bijective encoding $\mathsf{INJ} : \mathbb{G} \to \mathbb{Z}_p$ and does not have to rely on expensive number-

theoretic constructions of provably secure TCR functions. Second, one only needs one subgroup membership test for decryption, whereas the KD-scheme needs two. Depending on the underlying group such subgroup membership tests may be as expensive as one exponentiation. Third, the class of symmetric encryption schemes our KEM can be securely instantiated with is larger since we do not require authenticated encryption. This in particular makes it possible to rely on free redundancy-free "one-pass" symmetric techniques (which process the message to be encrypted only once). For authenticated encryption there are only less efficient two-pass schemes freely available since all one-pass techniques are covered by patents [3].

Acknowledgments. We thank Michel Abdalla, Mihir Bellare, Yevgeniy Dodis, Martijn Stam, and Moti Yung for comments and suggestions. This research was partially supported by the research program Sentinels (`http://www.sentinels.nl`). Sentinels is being financed by Technology Foundation STW, the Netherlands Organization for Scientific Research (NWO), and the Dutch Ministry of Economic Affairs. Parts of this paper were written while the author was a visitor at University of California San Diego supported by a DAAD postdoc fellowship.

## References

1. M. Abdalla, M. Bellare, and P. Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In D. Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 143–158. Springer-Verlag, Berlin, Germany, Apr. 2001.
2. M. Bellare, T. Kohno, and V. Shoup. Stateful public-key cryptosystems: How to encrypt with one 160-bit exponentiation. In A. Juels, R. N. Wright, and S. Vimercati, editors, *ACM CCS 06*, pages 380–389. ACM Press, Oct. / Nov. 2006.
3. M. Bellare, P. Rogaway, and D. Wagner. The EAX mode of operation. In B. K. Roy and W. Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 389–407. Springer-Verlag, Berlin, Germany, Feb. 2004.
4. D. J. Bernstein. Pippenger's exponentiation algorithm. Available from `http://cr.yp.to/papers.html`, 2001.
5. D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer-Verlag, Berlin, Germany, May 2004.
6. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer-Verlag, Berlin, Germany, Aug. 2004.
7. D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM Journal on Computing*, 5(36):915–942, 2006.
8. D. Boneh and M. K. Franklin. Identity based encryption from the Weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.
9. X. Boyen, Q. Mei, and B. Waters. Direct chosen ciphertext security from identity-based techniques. In V. Atluri, C. Meadows, and A. Juels, editors, *ACM CCS 05*, pages 320–329. ACM Press, Nov. 2005.
10. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO'98*, volume 1462 of *LNCS*, pages 13–25. Springer-Verlag, Berlin, Germany, Aug. 1998.

11. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.
12. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
13. R. R. Farashahi, B. Schoenmakers, and A. Sidorenko. Efficient pseudorandom generators based on the DDH assumption. In *PKC 2007*, volume ???? of *LNCS*, pages ???–??? Springer-Verlag, 2007.
14. D. Hofheinz, J. Herranz, and E. Kiltz. The Kurosawa-Desmedt key encapsulation is not chosen-ciphertext secure. Cryptology ePrint Archive, Report 2006/207, 2006. `http://eprint.iacr.org/`.
15. D. Hofheinz and E. Kiltz. Concise Hybrid Encryption. Manuscript, 2006.
16. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In S. Halevi and T. Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 581–600. Springer-Verlag, Berlin, Germany, Mar. 2006.
17. E. Kiltz. On the limitations of the spread of an IBE-to-PKE transformation. In M. Yung, Y. Dodis, A. Kiayias, and T. Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 274–289. Springer-Verlag, Berlin, Germany, Apr. 2006.
18. E. Kiltz. Chosen-ciphertext secure key-encapsulation based on Gap Hashed Diffie-Hellman (full version). Cryptology ePrint Archive, 2007. `http://eprint.iacr.org/`.
19. K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 426–442. Springer-Verlag, Berlin, Germany, Aug. 2004.
20. T. Okamoto and D. Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In K. Kim, editor, *PKC 2001*, volume 1992 of *LNCS*, pages 104–118. Springer-Verlag, Berlin, Germany, Feb. 2001.
21. D. H. Phan and D. Pointcheval. About the security of ciphers (semantic security and pseudo-random permutations). In H. Handschuh and A. Hasan, editors, *SAC 2004*, volume 3357 of *LNCS*, pages 182–197. Springer-Verlag, Berlin, Germany, Aug. 2004.
22. N. Pippenger. On the evaluation of powers and related problems. In *Proceedings of FOCS 1976*, pages 258–263, 1976.
23. C. Rackoff and D. R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 433–444. Springer-Verlag, Berlin, Germany, Aug. 1992.
24. V. Shoup. Lower bounds for discrete logarithms and related problems. In W. Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer-Verlag, Berlin, Germany, May 1997.

## A Proof of Theorem 1

Suppose there exists a polynomial time adversary $\mathcal{A}$ that breaks the chosen-ciphertext security of the encapsulation scheme with (non-negligible) advantage $\mathbf{Adv}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)$ and makes at most $q$ decapsulation queries.

We show that there exists an adversary $\mathcal{B}$ that runs in time $\mathbf{Time}_{\mathcal{B}}(k) = \mathbf{Time}_{\mathcal{A}}(k) + O(q \cdot \mathbf{Time}_{\mathbb{G}}(k))$, (where $\mathbf{Time}_{\mathbb{G}}(k)$ is the time to perform a basic

operation in $\mathbb{G}$) and runs adversary $\mathcal{A}$ as a subroutine to solve a random instance of the GHDH problem with advantage

$$\mathbf{Adv}^{\text{ghdh}}_{\mathsf{Gen},\mathcal{B}}(k) \geq \mathbf{Adv}^{kem\text{-}\text{cca}}_{\mathcal{KEM},\mathcal{A}}(k)\,. \tag{2}$$

Now Eqn. (2) proves the Theorem.

We now give the description of adversary $\mathcal{B}$. Adversary $\mathcal{B}$ inputs an instance of the GHDH problem, i.e. $\mathcal{B}$ inputs the values $(1^k, \mathcal{HG}, \mathsf{H}, g, u = g^a, g^b, W)$. $\mathcal{B}$'s goal is to determine whether $W = \mathsf{H}(u^b)$ or $W \in \{0,1\}^l$ is a random bit string. Adversary $\mathcal{B}$ runs adversary $\mathcal{A}$ simulating its view as in the original KEM security experiment as follows:

**Key Generation & Challenge.** Initially adversary $\mathcal{B}$ picks a random value $d \in \mathbb{Z}_p^*$ and defines the target ciphertext

$$C^* = (c^*, \pi^*) \leftarrow (g^b, (g^b)^d)\,. \tag{3}$$

and the challenge key as $K^* = W$. We denote $t^* = \mathsf{INJ}(c^*)$ as the target tag (associated with the target ciphertext). The value $v$ from the public key $pk = (u, v)$ is defined as

$$v \leftarrow u^{-t^*} \cdot g^d\,. \tag{4}$$

Note that the public key is identically distributed as in the original KEM. With each ciphertext $C = (c, \pi)$ we associate a tag $t = \mathsf{INJ}(c)$. Recall that we call a ciphertext consistent if $\pi = (u^t v)^r$, where $r = \log_g c$. Note that the way the keys are setup for a consistent ciphertext we have

$$\pi = (u^t v)^r = (u^t u^{-t^*} g^d)^r = (u^r)^{t-t^*} \cdot c^d\,. \tag{5}$$

Given a consistent ciphertext $C = (c, \pi)$ with associated tag $t \neq t^*$ the session key $K = \mathsf{H}(c^x)$ can alternatively be computed by Eqn. (5) as

$$K = \mathsf{H}((\pi / c^d)^{(t-t^*)^{-1}})\,. \tag{6}$$

By Eqn. (5) and since $t^* = \mathsf{INJ}(c^*)$ the challenge ciphertext $C^* = (c^*, \pi^*) = (g^b, (g^b)^d) = (c^*, (c^*)^d)$ is a correctly generated ciphertext for randomness $b$. If $W = \mathsf{H}(u^b)$ then it follows by Eqn. (4) that $C^* = (g^b, (g^b)^d)$ is a correct ciphertext of key $K^* = W = \mathsf{H}(u^b)$, distributed as in the original experiment. On the other hand, when $W$ is uniform and independent in $\{0,1\}^l$ then $C^*$ is independent of $K^* = W$ in the adversary's view.

Adversary $\mathcal{B}$ runs $\mathcal{A}$ on input $(pk, K^*, C^*)$ answering to its queries as follows:

**Decryption queries.** The KEM decapsulation queries are simulated by $\mathcal{B}$ as follows: Let $C = (c, \pi)$ be an arbitrary ciphertext submitted to the decapsulation oracle $\mathsf{DecO}(sk, \cdot)$. First $\mathcal{B}$ performs a consistency check of the ciphertext, i.e. it checks (using the Diffie-Hellman oracle $\mathsf{DDHsolve}_{\mathbb{G}}(\cdot,\cdot,\cdot,\cdot)$) if $(g, u^t v, c, \pi)$ is a valid Diffie-Hellman tuple.[3]

[3] At this point the existence of a weak DDH oracle $\mathsf{DDHsolve}_{g,u}(\cdot,\cdot)$ for fixed $u$ is sufficient. This is since $(g, u^t v, c, \pi)$ is a valid Diffie-Hellman tuple iff $(g, u, c, (\pi/c^d)^{(t-t^*)^{-1}})$ is a valid Diffie-Hellman tuple. So to verify consistency of the KEM ciphertext, $\mathcal{B}$ equivalently queries $\mathsf{DDHsolve}_{g,u}(c, (\pi/c^d)^{(t-t^*)^{-1}})$.

We remark that this is the only case where the simulation depends on the existence of the DDH oracle $\mathsf{DDHsolve}$. If $C$ is not consistent then $\mathcal{B}$ returns $\perp$. Otherwise, if the ciphertext is consistent $\mathcal{B}$ computes $t = \mathsf{INJ}(c)$ and distinguishes the following three cases:

**Case 1:** $t = t^*$ and $c = c^*$: adversary $\mathcal{B}$ rejects the query. In this case consistency (Eqn. (5)) implies $\pi = c^d = (c^*)^d = \pi^*$ and hence $C = C^*$ and the query made by $\mathcal{A}$ is illegal. Therefore it may be rejected by $\mathcal{B}$.

**Case 2:** $t = t^*$ and $c \neq c^*$: this is not possible since $\mathsf{INJ} : \mathbb{G} \to \mathbb{Z}_p$ is an injection. (If more generally we use a TCR function then at this point adversary $\mathcal{B}$ found a collision $c \neq c^*$ in $\mathsf{TCR}$ with $\mathsf{TCR}(c) = \mathsf{TCR}(c^*)$.)

**Case 3:** $t \neq t^*$: adversary $\mathcal{B}$ computes the correct session key by Eqn. (6) as $K \leftarrow \mathsf{H}((\pi/c^d)^{(t-t^*)^{-1}})$.

This completes the description of the decapsulation oracle.

We have shown that the simulation of the decapsulation oracle is always perfect, i.e. the output of the simulated decapsulation oracle is identically distributed as the output of $\mathsf{Dec}(sk, C)$.

**Guess.** Eventually, $\mathcal{A}$ outputs a guess $\delta' \in \{0,1\}$ where $\delta' = 1$ means that $K^*$ is the correct key. Algorithm $\mathcal{B}$ concludes its own game by outputting $\gamma' := \delta'$ where $\gamma' = 1$ means that $W = \mathsf{H}(g^{ab})$ (i.e. $\gamma = 1$) and $\gamma' = 0$ means that $W$ is random ($\gamma = 0$).

This completes the description of adversary $\mathcal{B}$.

Analysis. Define "F" to be the event that $\mathcal{B}$ wins its GHDH game, i.e. it outputs $\delta' = 1$ if $W = \mathsf{H}(g^{ab})$ and $\delta' = 0$ if $W$ is random. On the one hand, if $W$ is uniform and independent in $\{0,1\}^l$ then the challenge ciphertext $C^*$ is independent of $K^* = W$ in the adversary's view. In that case we have $\Pr[\mathrm{F}] = \Pr[\delta' = 0] = \frac{1}{2}$. On the other hand, when $W = \mathsf{H}(g^{ab})$ then $C^*$ is a correct ciphertext of the challenge key $K^*$, distributed as in the original experiment. Then, by our assumption, $\mathcal{A}$ must make a correct guess $\delta' = 1$ with advantage at least $\mathbf{Adv}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)$ and we have $|\Pr[\mathrm{F}] - \frac{1}{2}| = |\Pr[\delta' = 1] - \frac{1}{2}| \geq \mathbf{Adv}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)$.

Therefore, adversary $\mathcal{B}$'s advantage in the GHDH game is $\mathbf{Adv}^{\mathrm{ghdh}}_{\mathsf{Gen},\mathcal{B}}(k) \geq \mathbf{Adv}^{kem\text{-}\mathrm{cca}}_{\mathcal{KEM},\mathcal{A}}(k)$ which proves Eqn. (2) and completes the proof of the theorem.