# Decryptable Searchable Encryption

Thomas Fuhr[1] and Pascal Paillier[2]

[1] Direction Centrale de la Sécurité des Systèmes d'Information
thomas.fuhr@sgdn.pm.gouv.fr
[2] Cryptography & Innovation, Gemalto Security Labs
pascal.paillier@gemalto.com

**Abstract.** As such, public-key encryption with keyword search (a.k.a PEKS or searchable encryption) does not allow the recipient to decrypt keywords *i.e.* encryption is not invertible. This paper introduces searchable encryption schemes which enable decryption. An additional feature is that the decryption key and the trapdoor derivation key are *totally independent*, thereby complying with many contexts of application. We put forward a seemingly optimal construction for decryptable searchable encryption which makes use of one KEM, one IDKEM and a couple of hash functions. We define a proper security model for decryptable searchable encryption and show that basic security requirements on the underlying KEM and IDKEM are enough for our generic construction to be strongly secure in the random oracle model.

## 1 Introduction

*Background.* Among the most recent developments of public-key cryptography, the mechanisms for ID-based encryption [19,5,6,13,3] and public-key encryption with keyword search (PEKS) have become increasingly attractive thanks to their connections with many other (still unsolved) design issues. It seems that the idea of encryption with keyword search, also known as *searchable encryption* [4], appeared as a natural application of what one could achieve with bilinear maps, which already provided the basis for ID-based encryption. A more recent work [1] shows that these mechanisms are intimately related in the sense that they are induced by a common primitive known as an anonymous IDKEM [8].

Informally, a searchable encryption $c$ of a keyword $w$ can only be tested by the recipient who uses her private key to detect whether $c$ matches $w$ or not. This ability is transferrable to anyone under the form of a keyword-specific trapdoor $\mathsf{T}(w)$ which enables the search for encryptions of $w$. In a typical application of searchable encryption, the entity holding $\mathsf{T}(w)$ receives lots of encrypted keywords and filters out encryptions of $w' \neq w$. Searchable encryption, as currently achieved, does not require ciphertexts to be decryptable.

*Our Contributions.* This paper introduces searchable encryption schemes that enable decryption. We mention that the decryption key and the trapdoor derivation key are independent of each other, thereby complying with various contexts

W. Susilo, J.K. Liu, and Y. Mu. (Eds.): ProvSec 2007, LNCS 4784, pp. 228–236, 2007.

of application. We put forward a generic construction for decryptable searchable encryption. To achieve our goal, we make use of generic cryptographic primitives such as key encapsulation mechanisms (a.k.a. KEMs) [10] and identity-based versions of KEMs (IDKEMs). Our construction also employs a couple of hash functions. We define a proper security model for decryptable searchable encryption and investigate under which security requirements on the underlying KEM and IDKEM blocks our construction yields a maximally secure scheme. All security proofs considered in this paper stand in the random oracle model.

*Applications of Our Work.* Decryptable searchable encryption (DSE for short) extends the notion of PEKS and may therefore be used in every single application of PEKS. We may also find applications in the management of encrypted databases. In particular, since the decryption key and the trapdoor derivation key are generated independently from one another, data can be decrypted by an entity and trapdoors be generated by some other party. An illustrative example of this feature is as follows. Assume Bob is a telephone operator, Alice a subscriber, Charlie a state agency and Daniel a police inspector whose role consists in identifying subscribers belonging to the Mafia. Assume that Bob stores Alice's telephone statement encrypted with DSE, and that the decryption key belongs to Alice and the trapdoor derivation key belongs to Charlie. Alice is the only person who can decrypt it, but Charlie can issue trapdoors for some phone numbers and give them to Daniel to help him find out whether Alice is connected to the Mafia, without learning anything about the other numbers Alice has called. The same scenario is applicable to the secure management of money transfers, wherein a maximal level of secrecy about account numbers involved in transactions is guaranteed, while leaving to a designated authority the ability to trace encrypted transactions made to or from well-identified bank accounts.

*Outline.* We start in Section 2 by a number of definitional facts about KEMs and IDKEMs. Section 3 describes our generic construction and provides a security analysis. Section 4 provides an example of instantiation based on ElGamal and $\mathsf{BDOP}$ [4]. Section 5 concludes on a number of questions left open by this work.

## 2 Preliminaries on Encapsulation Mechanisms

### 2.1 Key Encapsulation Mechanisms (KEMs)

*Definition.* A KEM is a basic cryptographic primitive by the means of which one can publicly and securely encapsulate a randomly generated session key. The owner of the private key (the decapsulation key) can later recover the session key given the encapsulation. KEMs make use of decapsulation keys, encapsulation keys, random numbers, ciphertexts and secret values (that may be symmetric keys). Here we will not describe their inner structure, but rather give a general description of the primitive. We identify a KEM to a tuple of probabilistic algorithms $\mathsf{KEM} = (\mathsf{KEM.Gen}, \mathsf{KEM.Encap}, \mathsf{KEM.Decap})$ defined as follows.

**Key generation.** $\mathsf{KEM.Gen}(1^k)$ takes a security parameter $k \in \mathbb{N}$ and outputs a public key $\mathsf{pk}_\mathrm{K} \in \mathsf{KEM}.\mathcal{PK}$ and a decapsulation key $\mathsf{sk}_\mathrm{K} \in \mathsf{KEM}.\mathcal{SK}$.

**Encapsulation.** $\mathsf{KEM.Encap}(\mathsf{pk}_\mathrm{K}, r)$ takes as input a public key $\mathsf{pk}_\mathrm{K}$ and a random $r \in \mathsf{KEM}.\mathcal{R}$ and returns an encapsulation $c \in \mathsf{KEM}.\mathcal{C}$ and the encapsulated value $s \in \mathsf{KEM}.\mathcal{S}$.

**Decapsulation.** $\mathsf{KEM.Decap}(\mathsf{sk}_\mathrm{K}, c)$ takes as input a decapsulation key $\mathsf{sk}_\mathrm{K}$ and an encapsulation $c$ and returns the matching decapsulated value $s \in \mathsf{KEM}.\mathcal{S}$.

It is well-known that the notion of KEM is equivalent to the one of public-key encryption. There has been a recent interest in lightening the relations between hybrid encryption and various forms of key encapsulations [2].

*Security Notions for KEMs.* As for other cryptographic primitives, one may define several security notions for KEMs. In particular, active attacks are defined similarly to chosen-ciphertext attacks against public-key encryption schemes. In this work, we mainly take interest in two security notions for KEMs which we describe under the form of games. The first security notion captures the property that the encapsulation function $\mathsf{KEM.Encap}$ cannot be inverted under an active attack:

**Game 1** ($\mathsf{r\text{-}OW\text{-}CCA.KEM}$)**.** *A probabilistic algorithm $\mathcal{A}$ is given a random key pair $(\mathsf{pk}_\mathrm{K}, \mathsf{sk}_\mathrm{K}) \leftarrow \mathsf{KEM.Gen}(1^k)$ as well as random $(c^*, s^*)$ and attempts to recover $r^* \in \mathsf{KEM}.\mathcal{R}$ such that $\mathsf{KEM.Encap}(\mathsf{pk}_\mathrm{K}, r^*) = (c^*, s^*)$.*

Unless otherwise stated, we denote by $\mathbf{Succ}(\mathcal{A}, k)$ the probability (taken over the random coins of $\mathcal{A}$ and its challenger) under which $\mathcal{A}$ wins a given security game. For any security notion $\mathsf{SEC}$ for any cryptographic primitive $\mathsf{PRIM}$ defined by such a game, we define $\mathbf{InSec}(\mathsf{SEC.PRIM}, k) = \max_{\mathcal{A}} \mathbf{Succ}(\mathcal{A}, k)$ where the maximum is taken over all polynomial time adversaries $\mathcal{A}$ playing the above game. $\mathsf{PRIM}$ is said to be $\mathsf{SEC}$-secure if $\mathbf{InSec}(\mathsf{SEC.PRIM}, k)$ is a negligible function of $k$.

The second security notion also relates to active attacks. It is quite similar to plaintext-checking attacks against public-key cryptosystems. It states that the decapsulation procedure is hard to compute without the decapsulation key, even when one is given an oracle that tells when the wanted decapsulated value is found.

**Game 2** ($\mathsf{s\text{-}OW\text{-}PCA.KEM}$)**.** *A probabilistic algorithm $\mathcal{A}$ is given $\mathsf{pk}_\mathrm{K}$ where $(\mathsf{pk}_\mathrm{K}, \mathsf{sk}_\mathrm{K}) \leftarrow \mathsf{KEM.Gen}(1^k)$ as well as a random encapsulation $c^*$, and attempts to recover the decapsulated value $s^*$ matching $c^*$. During the game, the adversary $\mathcal{A}$ is given access to a distinguisher (or distinction oracle) which, given a pair $(c, s)$, tells whether $c$ encapsulates $s$. The oracle can be invoked without restrictions by $\mathcal{A}$.*

### 2.2 Identity-Based Key Encapsulation Mechanisms (IDKEM)

*Definition.* An IDKEM is an identity-based KEM. Definitionally, IDKEMs can be defined as *searchable* KEMs, a primitive providing the trapdoor mechanism

underlying searchable encryption. Because this aspect of IDKEMs is important with regard to this work, we make use of the widest definition. An IDKEM is identified to a tuple of probabilistic algorithms $\mathsf{IDKEM} = (\mathsf{IDKEM.Gen}, \mathsf{IDKEM.Trap}, \mathsf{IDKEM.Encap}, \mathsf{IDKEM.Decap})$ defined as follows.

**Key generation.** $\mathsf{IDKEM.Gen}(1^k)$ takes a security parameter $k$ and outputs a public key $\mathsf{pk_I} \in \mathsf{IDKEM}.\mathcal{PK}$ and a trapdoor derivation key $\mathsf{tk_I} \in \mathsf{IDKEM}.\mathcal{TK}$.

**Trapdoor derivation.** $\mathsf{IDKEM.Trap}(\mathsf{tk_I}, w)$ makes use of the trapdoor derivation key to compute a decapsulation trapdoor $\mathsf{T}(w) \in \mathsf{IDKEM}.\mathcal{T}$ for the keyword $w \in \{0,1\}^{\mathsf{w}}$. Trapdoor derivation may be probabilistic.

**Encapsulation.** $\mathsf{IDKEM.Encap}(\mathsf{pk_I}, w, r)$ takes a public key, a keyword $w \in \{0,1\}^{\mathsf{w}}$ and a random $r \in \mathsf{IDKEM}.\mathcal{R}$ and outputs an encapsulation $c \in \mathsf{IDKEM}.\mathcal{C}$ and the encapsulated value $u \in \mathsf{IDKEM}.\mathcal{U}$.

**Decapsulation.** $\mathsf{IDKEM.Decap}(\mathsf{T}(w), c)$ takes an encapsulation $c$ and a decapsulation trapdoor $\mathsf{T}(w)$ and returns the decapsulated value $u$ matching $w$ and $c$.

*Security Notions.* As in the case of ID-based encryption schemes, security notions come in two different flavors for IDKEMs. A first family of adversarial goals captures different levels of privacy with respect to the decapsulated value (one-wayness, indistinguishability, etc.). The others are defined in a similar way but relate to the privacy of the keyword $w$ itself, and resistance to these goals is identified as a form of anonymity.

**Game 3** ($\mathsf{s\text{-}OW\text{-}CCA.IDKEM}$)**.** *The adversary $\mathcal{A}$ is given a public key $\mathsf{pk_I}$ where $(\mathsf{pk_I}, \mathsf{tk_I}) \leftarrow \mathsf{IDKEM.Gen}(1^k)$ and later outputs a keyword $w^+ \in \{0,1\}^{\mathsf{w}}$. The challenger randomly selects $r^+ \leftarrow \mathsf{IDKEM}.\mathcal{R}$ and computes $(c^+, u^+) = \mathsf{IDKEM.Encap}(\mathsf{pk_I}, w^+, r^+)$. The challenge $c^+$ is sent to $\mathcal{A}$ and $\mathcal{A}$ attempts to recover $u^+$. Throughout the game, the adversary is given access to two oracles; a distinction oracle which tells whether a tuple $(w, c, u)$ is consistent in the sense that $c$ encapsulates $u$ under keyword $w$; the adversary has also access to a trapdoor derivation oracle but is not allowed to request a trapdoor corresponding to $w^+$.*

**Game 4** ($\mathsf{ANON\text{-}CCA.IDKEM}$)**.** *The adversary $\mathcal{A}$ is given a public key $\mathsf{pk_I}$ where $(\mathsf{pk_I}, \mathsf{tk_I}) \leftarrow \mathsf{IDKEM.Gen}(1^k)$ and later returns two different keywords $w_0, w_1 \in \{0,1\}^{\mathsf{w}}$. The challenger picks a random bit $b$, randomly selects $r \leftarrow \mathsf{IDKEM}.\mathcal{R}$ and computes $(c_b, u_b) = \mathsf{IDKEM.Encap}(\mathsf{pk_I}, w_b, r)$. The encapsulation $c_b$ is sent to the adversary. The adversary later outputs a guess $\hat{b}$ and wins the game if $\hat{b} = b$. During the game, the adversary is allowed to query two oracles: a trapdoor derivation oracle for $w \neq w_0, w_1$ and a decapsulation oracle for $w \in \{w_0, w_1\}$ and $c \neq c_b$ which returns the decapsulation $u$ of $c$. Here $\mathbf{Succ}(\mathcal{A}, k)$ is defined as the difference between the probability that $\mathcal{A}$ wins the game and $1/2$, the probability for a random response to be true.*

## 3 Decryptable Searchable Encryption

### 3.1 Definition and Security Model

We identify a decryptable searchable encryption scheme $\mathsf{DSE}$ to a tuple of probabilistic algorithms

$$\mathsf{DSE} = (\mathsf{DSE.Gen}, \mathsf{DSE.Enc}, \mathsf{DSE.Dec}, \mathsf{DSE.Trap}, \mathsf{DSE.Test})$$

enjoying the following properties.

**Key generation.** $\mathsf{DSE.Gen}(1^k)$ takes a security parameter $k$ and outputs a public key $\mathsf{pk}$, a decryption key $\mathsf{dk}$ and a trapdoor derivation key $\mathsf{tk}$.

**Encryption.** $\mathsf{DSE.Enc}(\mathsf{pk}, w, r)$ takes as input a public key $\mathsf{pk}$, a message $w \in \{0,1\}^{\mathsf{w}}$ and $r \in \mathsf{DSE}.\mathcal{R}$ and returns a ciphertext $c$.

**Decryption.** $\mathsf{DSE.Dec}(\mathsf{dk}, c)$ takes a decryption key $\mathsf{dk}$ and a ciphertext $c$ and returns the message $w \in \{0,1\}^{\mathsf{w}}$ that $c$ encrypts or $\perp$ if $c$ is invalid.

**Trapdoor derivation.** $\mathsf{DSE.Trap}(\mathsf{tk}, w)$ requires a trapdoor derivation key $\mathsf{tk}$ and a message $w \in \{0,1\}^{\mathsf{w}}$ to compute a search trapdoor $\mathsf{T}(w)$. The trapdoor $\mathsf{T}(w)$ may be probabilistic in which case $\mathsf{DSE.Trap}$ also requires randomness.

**Test.** $\mathsf{DSE.Test}(\mathsf{pk}, \mathsf{T}(w), c)$ takes as input a public key $\mathsf{pk}$, a search trapdoor $\mathsf{T}(w)$ for $w$, a ciphertext $c$ and returns 1 if $c$ encrypts $w$. Otherwise 0 is returned.

We focus on the strongest possible notion of security for decryptable searchable encryption which we capture by the following game:

**Game 5** ($\mathsf{IND\text{-}CCA.DSE}$). *A set of keys* $(\mathsf{pk}, \mathsf{dk}, \mathsf{tk}) \leftarrow \mathsf{DSE.Gen}(1^k)$ *is randomly selected and* $\mathcal{A}$ *is executed over* $\mathsf{pk}$. $\mathcal{A}$ *outputs* $w_0, w_1 \in \{0,1\}^{\mathsf{w}}$ *with* $w_0 \neq w_1$. *The challenger randomly picks* $b \in \{0,1\}$ *and outputs* $c_b$. $\mathcal{A}$ *then outputs its guess* $\hat{b} \in \{0,1\}$ *and wins if* $\hat{b} = b$. *Throughout the game,* $\mathcal{A}$ *may send queries* $w \notin \{w_0, w_1\}$ *to a trapdoor derivation oracle* $\mathsf{DSE.Trap}(\mathsf{tk}, \cdot)$ *and queries* $c \neq c_b$ *to a decryption oracle* $\mathsf{DSE.Dec}(\mathsf{dk}, \cdot)$.

### 3.2 A Generic Construction from KEMs and IDKEMs

We suggest a construction using one KEM, one IDKEM and a couple of hash functions $H_1, H_2$. These ingredients have to be *compatible* in the sense that $H_1$ must map elements of $\mathsf{IDKEM}.\mathcal{U}$ to elements of $\mathsf{KEM}.\mathcal{R}$, and elements of $\mathsf{KEM}.\mathcal{S}$ to elements of $\{0,1\}^{\mathsf{w}} \times \mathsf{IDKEM}.\mathcal{R}$. We also require an additional property for the IDKEM:

*Property 1.* Given $(w_0, w_1) \in \{0,1\}^{\mathsf{w}}$ and $(c_0, c_1) \in \mathsf{IDKEM}.\mathcal{C}$, it is easy to check whether for some $s \in \mathsf{IDKEM}.\mathcal{S}$, the first component $c$ of

$$(c, u) = \mathsf{IDKEM.Encap}(\mathsf{pk}_{\mathrm{I}}, w_i, s)$$

is equal to $c_i$ for $i = 0, 1$, this holding for any public key $\mathsf{pk}_{\mathrm{I}}$.

This property is achieved for most existing IDKEMs. We denote by $\odot$ a revertible composition law over $\mathsf{IDKEM}.\mathcal{R}$ such as a group law or $\oplus$. We define our construction $\mathsf{DSE}$ as follows.

**Key generation.** $\mathsf{DSE.Gen}(1^k)$ runs $\mathsf{KEM.Gen}(1^k)$ and $\mathsf{IDKEM.Gen}(1^k)$ and sets $\mathsf{pk} = (\mathsf{pk}_\mathrm{K}, \mathsf{pk}_\mathrm{I})$, $\mathsf{dk} = \mathsf{sk}_\mathrm{K}$ and $\mathsf{tk} = \mathsf{tk}_\mathrm{I}$.

**Encryption.** $\mathsf{DSE.Enc}(\mathsf{pk}, w, r)$ runs $\mathsf{KEM.Encap}(\mathsf{pk}_\mathrm{K}, r)$ resulting in an encapsulation $c_1$ and a decapsulated value $s$. One computes $(s_1, s_2) = H_1(s)$, $c_2 = s_1 \oplus w$, $(c_3, u) = \mathsf{IDKEM.Encap}(\mathsf{pk}_\mathrm{I}, w, s_2)$ and $c_4 = r \odot H_2(u)$. The ciphertext is $(c_1, c_2, c_3, c_4)$.

**Decryption.** $\mathsf{DSE.Dec}(\mathsf{dk}, (c_1, c_2, c_3, c_4))$ first decapsulates $c_1$ by running $\mathsf{KEM.Decap}(\mathsf{sk}_\mathrm{K}, c_1)$ to recover $s$. One then computes $(s_1, s_2) = H_1(s)$, $w = s_1 \oplus c_2$, and $(c', u) = \mathsf{IDKEM.Encap}(\mathsf{pk}_\mathrm{I}, w, s_2)$. The algorithm checks that $c' = c_3$, computes $r' = H_2(u)^{-1} \odot c_4$ and finally checks that $\mathsf{KEM.Encap}(\mathsf{pk}_\mathrm{K}, r') = (c_1, s)$ before returning $w$. In case one of these conditions is not fulfilled, $\perp$ is returned.

**Trapdoor derivation.** $\mathsf{DSE.Trap}(\mathsf{tk}, w)$ returns $\mathsf{IDKEM.Trap}(\mathsf{tk}_\mathrm{I}, w)$.

**Test.** $\mathsf{DSE.Test}(\mathsf{pk}, \mathsf{T}(w), (c_1, c_2, c_3, c_4))$ computes

$$u' = \mathsf{IDKEM.Decap}(\mathsf{T}(w), c_3)\ , \quad r' = c_4 \odot H(u')^{-1}\ ,$$

$(c_1', s') = \mathsf{KEM.Encap}(\mathsf{pk}_\mathrm{K}, r')$ and $(s_1, s_2) = H_1(s')$. One then checks whether $c_1' = c_1$ and $(c_3, u) = \mathsf{IDKEM.Encap}(\mathsf{pk}_\mathrm{I}, c_2 \oplus s_1, s_2)$. $\mathsf{DSE.Test}$ returns 1 if these conditions are fulfilled, 0 otherwise.

**Theorem 1.** *Assuming that* $\mathsf{KEM}$ *is* $\mathsf{r\text{-}OW\text{-}CCA}$ *and* $\mathsf{s\text{-}OW\text{-}PCA}$*-secure and that* $\mathsf{IDKEM}$ *is* $\mathsf{s\text{-}OW\text{-}CCA}$ *and* $\mathsf{ANON\text{-}CCA}$*-secure,* $\mathsf{DSE}$ *as per the above construction is* $\mathsf{IND\text{-}CCA}$*-secure. More precisely,*

$$\begin{aligned}\mathbf{InSec}(\mathsf{IND\text{-}CCA.DSE}, k) &\leq 2 \cdot \mathbf{InSec}(\mathsf{s\text{-}OW\text{-}PCA.KEM}, k) \\ &+ 2 \cdot \mathbf{InSec}(\mathsf{s\text{-}OW\text{-}CCA.IDKEM}, k) + \mathbf{InSec}(\mathsf{r\text{-}OW\text{-}CCA.KEM}, k) \\ &+ \mathbf{InSec}(\mathsf{ANON\text{-}CCA.IDKEM}, k) + \mathsf{negl}\,(k)\ .\end{aligned}$$

We give a full proof of Theorem 1 in the full version of this work [12].

## 4 An Efficient Instantiation of DSE Using Bilinear Maps

We now give a specific scheme using our general construction. To this end, we will use existing examples of KEMs and IDKEMs.

### 4.1 Description of Our Scheme

We now consider the decryptable searchable encryption scheme as per our construction of Section 3 using $\mathsf{IDKEM} = \mathsf{BDOP}$ [4] over a bilinear group system

$\mathcal{S} = (\mathbb{G}_1, g_1, \mathbb{G}_2, g_2, \mathbb{G}_t, e(\cdot,\cdot))$ and a $\mathsf{KEM}$ defined over $\mathbb{G}_1$ and relying on ElGamal encryption [11]. It is easily seen that $\mathsf{BDOP}$ satisfies Property 1. The revertible operator $\odot$ is taken as addition modulo $q = |\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_t|$. The security parameter $k$ is set to $\log q$. We also employ three hash functions $F, G, H$ viewed as random oracles which domain and range are implicitly defined by the following description.

**Key generation.** $\mathsf{DSE.Gen}$ randomly selects $x \leftarrow \mathbb{Z}_q$ and $x' \leftarrow \mathbb{Z}_q$ and sets the decryption key to $\mathsf{dk} = x$ and the trapdoor derivation key to $\mathsf{tk} = x'$. Noting $y = g_1^x$, $y' = g_1^{x'}$, the public key is $\mathsf{pk} = (y, y')$.

**Encryption.** For $w \in \{0,1\}^{\mathsf{w}}$ and $r \in \mathbb{Z}_q$, $\mathsf{DSE.Enc}(\mathsf{pk}, w, r)$ computes $c_1 = g_1^r$, $(s_1, s_2) = G(y^r)$, $c_2 = s_1 \oplus w$, $c_3 = g_1^{s_2}$, $u = e(y'^{s_2}, F(w))$ and $c_4 = H(u) + r \bmod q$. The encryption is $c = (c_1, c_2, c_3, c_4)$.

**Decryption.** Given $c = (c_1, c_2, c_3, c_4)$, $\mathsf{DSE.Dec}(x, c)$ computes $s = c_1^x$, $(s_1, s_2) = G(s)$ and $w = c_2 \oplus s_1$. If $c_3 \neq g_1^{s_2}$, $\mathsf{DSE.Dec}(x, c)$ returns $\perp$. Otherwise, one computes $u = e(y'^{s_2}, F(w))$, $r = c_4 - H(u) \bmod q$ and checks whether $c_1 = g_1^r$. If this condition is satisfied, $\mathsf{DSE.Dec}(x, c)$ returns $w$. Otherwise $\perp$ is returned.

**Trapdoor derivation.** Given $w \in \{0,1\}^{\mathsf{w}}$ and $x'$, $\mathsf{DSE.Trap}(x', w)$ returns $\mathsf{T}(w) = F(w)^{x'} \in \mathbb{G}_2$.

**Test.** Given $c = (c_1, c_2, c_3, c_4)$ and $\mathsf{T}(w) \in \mathbb{G}_2$, $\mathsf{DSE.Test}(\mathsf{pk}, \mathsf{T}(w), c)$ computes $u = e(c_3, \mathsf{T}(w))$ and $r = c_4 - H(u) \bmod q$. If $c_1 \neq g_1^r$, 0 is returned. Otherwise one computes $s = y^r$, $(s_1, s_2) = G(s)$ and $w = c_2 \oplus s_1$. If $c_3 \neq g_1^{s_2}$, 0 is returned. Otherwise 1 is returned.

### 4.2 Security Analysis

*The Gap-Diffie-Hellman Problem* $\mathsf{GDH}$. Let $\mathbb{G}$ be a group of prime order $q$ and let $g$ be a generator of $\mathbb{G}$. The computational problem $\mathsf{CDH}$ is defined as the problem of computing $g^{ab}$ given $(g^a, g^b) \in \mathbb{G}$. $\mathsf{DDH}$ consists in distinguishing the two distributions $D = (g^a, g^b, g^{ab})$ and $R = (g^a, g^b, g^t)$ for randomly selected $a, b, t \leftarrow \mathbb{Z}_q$. The gap problem $\mathsf{GDH}$ is defined as the problem of solving $\mathsf{CDH}$ given an oracle that solves $\mathsf{DDH}$. These problems are classical in cryptography and we refer the reader to an extensive literature [9,18,14,7] for applications of $\mathsf{GDH}$ to public-key design. We finally note that $\mathsf{GDH} \equiv \mathsf{CDH}$ over bilinear map groups.

*The Gap-Bilinear-Diffie-Hellman Problem* $\mathsf{GBDH}$. Let $\mathcal{S}$ be a bilinear group system as above. The computational problem $\mathsf{CBDH}$ is defined as the problem of computing $e(g_1, g_2)^{abc}$ given $g_1^a, g_1^b \in \mathbb{G}_1$ and $g_2^c \in \mathbb{G}_2$. $\mathsf{CBDH}$ admits a decisional version $\mathsf{DBDH}$ which consists in distinguishing the two distributions $D = (g_1^a, g_1^b, g_2^c, e(g_1, g_2)^{abc})$ and $R = (g_1^a, g_1^b, g_2^c, e(g_1, g_2)^t)$ for randomly selected $a, b, c, t \leftarrow \mathbb{Z}_q$. It is easily shown that $\mathsf{DBDH} \Leftarrow \mathsf{CBDH}$, which allows one to define the gap problem $\mathsf{GBDH}$ as the problem of solving $\mathsf{CBDH}$ given an oracle that solves $\mathsf{DBDH}$.

We now claim the following facts: *a)* ElGamal (defined over the group $\mathbb{G}_1$) is s-OW-PCA-secure under the assumption that GDH is intractable over $\mathbb{G}_1$; *b)* it is also r-OW-CCA-secure under the discrete log assumption over $\mathbb{G}_1$; *c)* one has $\mathbf{InSec}(\text{ANON-CCA.BDOP}, k) = 0$ for any $k \in \mathbb{N}$; *d)* BDOP is s-OW-CCA-secure under the assumption that GBDH is intractable over $\mathcal{S}$ in the random oracle model; *e)* the GBDH problem over $\mathcal{S}$ is reducible to the GDH problem over $\mathbb{G}_1$; *f)* DSE[ElGamal, BDOP] is IND-CCA-secure under the GBDH assumption over $\mathcal{S}$ and the GDH assumption over $\mathbb{G}_1$ in the random oracle model.

As a direct application of the above, we state:

**Theorem 2.** DSE[ElGamal, BDOP] *is* IND-CCA*-secure under the* GDH *assumption over* $\mathbb{G}_1$ *in the random oracle model.*

We refer to the full version of this work [12] for more details and proper proofs of these statements.

## 5 Conclusion and Open Issues

We introduced the concept of decryptable searchable encryption and showed how to generically implement this new primitive using one KEM, one IDKEM and hash functions. We provided a precise security proof in which we relate IND-CCA-security to the security properties of the inner primitives. Decryptable searchable encryption finds applications in the secure management of encrypted databases, among others. We mention that ID-based decryptable searchable encryption is obtained as a side effect of our generic construct. This is done by replacing the underlying KEM by a second IDKEM which keyword input is fed with ID strings. The trapdoor derivation key of this IDKEM is then the master secret key of the whole ID-based encryption scheme.

We see several open research topics associated to our work. First, one may ask whether more efficient constructions exist that achieve general-purpose DSE. Optimizing the ciphertext size is a pending issue in this respect. Second, we would consider as a major breakthrough to come up with a DSE which security does not rely on random oracles. Although seemingly hard to find, such a scheme would benefit from a security standing in the standard model and would consequently avoid the threat of recent separation results [15,16,17].

## References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 205–222. Springer, Heidelberg (2005)
2. Abe, M., Gennaro, R., Kurosawa, K., Shoup, V.: Tag-kem/dem: A new framework for hybrid encryption and a new analysis of the kurosawa-desmedt kem. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 426–442. Springer, Heidelberg (2005)

3. Boneh, D., Boyen, X., Goh, E.-J.: Hierarchical identity based encryption with constant size ciphertext. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 440–456. Springer, Heidelberg (2005)
4. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
5. Boneh, D., Franklin, M.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
6. Boyen, X.: Multipurpose identity-based signcryption (a swiss army knife for identity-based cryptography). In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003)
7. Cha, J.C., Cheon, J.H.: An identity-based signature from gap diffie-hellman groups. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 18–30. Springer, Heidelberg (2002)
8. Chen, L., Cheng, Z., Malone-Lee, J., Smart, N.P.: An efficient ID-KEM based on the sakai-kasahara key construction. In: Cryptology ePrint Archive, Report 2005/224 (2005), `http://eprint.iacr.org/`
9. Coron, J.-S., Handschuh, H., Joye, M., Paillier, P., Pointcheval, D., Tymen, C.: Optimal chosen-ciphertext secure encryption of arbitrary-length messages. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 17–33. Springer, Heidelberg (2002)
10. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen-ciphertext attacks. SIAM Journal on Computing 33(1), 167–226 (2003)
11. ElGamal, T.: A public key cryptosystem and signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31, 469–472 (1985)
12. Fuhr, T., Paillier, P.: Decryptable searchable encryption. Cryptology ePrint Archive (2007), `http://eprint.iacr.org/`
13. Gentry, C., Silverberg, A.: Hierarchical ID-based cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)
14. Okamoto, T., Pointcheval, D.: The gap-problems: A new class of problems for the security of cryptographic schemes. In: Kim, K.-c. (ed.) PKC 2001. LNCS, vol. 1992, pp. 104–118. Springer, Heidelberg (2001)
15. Paillier, P., Vergnaud, D.: Discrete-log-based signatures may not be equivalent to discrete log. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 1–20. Springer, Heidelberg (2005)
16. Paillier, P., Villar, J.: Trading one-wayness against chosen-ciphertext security in factoring-based encryption. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 252–266. Springer, Heidelberg (2006)
17. Paillier, P.: Impossibility proofs for RSA signatures in the standard model. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 31–48. Springer, Heidelberg (2006)
18. Pointcheval, D.: Chosen-ciphertext security for any one-way cryptosystem. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 129–146. Springer, Heidelberg (2000)
19. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)