Refine the Concept of Public Key Encryption with Delegated Search

Qiang Tang[1] and Yuanjie Zhao[2] and Xiaofeng Chen[3] and Hua Ma[2]

[1] APSIA group, SnT, University of Luxembourg
6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg
qiang.tang@uni.lu

[2] Department of Applied Mathematics
Xidian University, Xian 710071, P.R. China
xdzhaoyuanjie@163.com and ma_hua@126.com

[3] State Key Laboratory of Integrated Service Networks (ISN)
Xidian University, Xian 710071, P.R. China
xfchen@xidian.edu.cn

November 12, 2012

Abstract. We revisit the concept of public key encryption with delegated keyword search (PKEDS), a concept proposed by Ibraimi et al. A PKEDS scheme allows a receiver to authorize third-party server(s) to search in two ways: either according to a message chosen by the server itself or according to a trapdoor sent by the receiver. We show that the existing formulation has some defects and the proposed scheme is unnecessarily inefficient. Based on our analysis, we present a refined formulation of the primitive with a new security model. We then propose a new PKEDS scheme, which is proven secure and much more efficient than the original scheme by Ibraimi et al.

1 Introduction

With the development of the internet, information security has become more and more important. One of the most effective technologies to protect information security is public-key encryption. In the public-key setting, Alice can encrypt an email with the public key of Bob, as a result, only Bob can learn the contents of the email. However, while encryption prevents an attacker from learning confidential information, it also helps an attacker to hide malicious contents.

Consider the following scenario: *Bob is an employee of the company X and all emails sent to Bob will be encrypted and stored in a server managed by X. When Bob wants to read his emails, he downloads the encrypted emails from the server and then*

decrypt them using his private key. Suppose Alice encrypts an email with the public key of Bob, but unfortunately, her computer is infected by some virus which embeds a malware into the email. In this case, the malware is also encrypted with the public-key of Bob. As the server managed by X is not able to scan for malicious codes with the encrypted emails, the malware can spread through this way. The first approach to detect encrypted malware is sending Bob's private key to the server. Given the private key of Bob, the server can decrypt the ciphertext and detect the malware. However, it also allows the server to read the plaintext emails. Another approach is to let Bob to scan his email for malicious contents. However, in this way, Bob's computer will be under the risk of infection by the malware. In addition, the task of Bob's may be too heavy.

In [23], the authors construct a public-key encryption scheme with delegated search(PKEDS) to deal with this problem. A PKEDS scheme possesses the following properties.

- Unlike most PEKS schemes [6], where only the matadata part (or, keywords) of the ciphertext is searchable, search directly happens in the ciphertext for PKEDS.
- If a server is delegated by Bob, then it can create any trapdoor without contacting Bob. Therefore, once delegation is done, Bob can go offline.
- The server can answer queries defined by Bob, who provides a message-dependent trapdoor associated with a specific word w. The server can test whether the word w equals to the plaintext of any ciphertext, without being able to recover w.

1.1 Contribution

Our contributions in this paper are threefold.

1. We analyse the PKEDS formulation and show some defects in the primitive formulation and some deficiency in the scheme construction, proposed in [23]. In particular, we argue that it is an undesirable feature that a search based on a message-dependent trapdoor will always require a master trapdoor as input. We also point out some flaws in their security analysis.sec:intro
2. We present a refined formulation for PKEDS and extend it to the multi-server setting. Correspondingly, we also extend the definition of trapdoor indistinguishability. Different that from [23], in our formulation, a search based on a message-dependent trapdoor does not require a master trapdoor as input. As a result, we can achieve better security against those servers which only perform searches based on message-dependent trapdoors.

3. Finally, we present a new construction for PKEDS and prove its security in our security model. The new construction is mainly based on ElGamal in bilinear groups, and a Boneh-Franklin IBE is employed because of its key privacy property. Compared with the original scheme from [23], the encryption algorithm becomes more complex due to the fact that a Boneh-Franklin ciphertext is added in order to get rid of the aforementioned undesirable feature. Nevertheless, the test algorithm based on master trapdoor is much more efficient compared to that in [23].

1.2 Organization

In Section 2, we review the PKEDS formulation and the proposed scheme (INHJ) in [23]. In Section 3, we refine the concept of PKEDS. In Section 4, we propose a new scheme and give a comparison with the INHJ scheme. In Section 5, we review the related work. In Section 6, we conclude the paper.

2 Review of the INHJ Scheme

In this section, we describe some observations on the PKEDS formulation, the proposed scheme, and the security proofs from [23]. These issues motivate us to refine the model and propose new efficient instantiation.

The INHJ scheme consists of the following nine algorithms, where the algorithms (rKeyGen, Encrypt, Decrypt) define an ElGamal encryption scheme.

1. Setup(1^λ): This algorithm outputs public parameters (pp) which contain the description of groups $< \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T >$ of prime order p, a bilinear map $\hat{e}$: $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, g_1 and g_2 as the generators of groups $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively.
2. sKeyGen(pp): Run by a server, this algorithm selects $x \in_R \mathbb{Z}_p$ and outputs the server's key pair:
$$(SK_s, PK_s) = (x,\ g_2^x).$$
3. rKeyGen(pp): Run by a receiver, this algorithm selects $y, \alpha \in_R \mathbb{Z}_p$ and outputs the receiver's private/public key pair:
$$(SK_r, PK_r) = ((y, g_2^\alpha),\ g_1^y).$$
4. Encrypt(PK_r, w): On input of the receiver's public key and a word $w \in \mathbb{G}_1$, this algorithm selects $k \in_R \mathbb{Z}_p$ and outputs the ElGamal ciphertext:
$$\begin{aligned} c_w &= (c_1, c_2) \\ &= (w \cdot (PK_r)^k,\ g_1^k) \\ &= (w \cdot g_1^{y \cdot k},\ g_1^k). \end{aligned}$$

5. $\mathsf{Delegate}(PK_s, SK_r)$: The algorithm creates a master trapdoor to let the server search the encrypted data for any word of his choice. The algorithm picks at random $r_1, r_2 \in \mathbb{Z}_p$ and outputs the master trapdoor:

$$\begin{aligned} t_* &= (t_1, t_2, t_3, t_4) \\ &= (g_2^{\alpha} \cdot (PK_s)^{r_1},\ g_2^{r_1},\ g_2^{y \cdot \alpha} \cdot (PK_s)^{r_2},\ g_2^{r_2}) \\ &= (g_2^{\alpha} \cdot g_2^{x \cdot r_1},\ g_2^{r_1},\ g_2^{y \cdot \alpha} \cdot g_2^{x \cdot r_2},\ g_2^{r_2}). \end{aligned}$$

6. $\mathsf{TrapGen}(SK_r, PK_s, w)$: The algorithm creates a trapdoor to let the server search for a specific massage w. The algorithm selects $\delta \in_R \mathbb{Z}_p$ and outputs the trapdoor:

$$\begin{aligned} t_w &= (t_5,\ t_6) \\ &= (\hat{e}(w, g_2^{\alpha}) \cdot \hat{e}(PK_r, (PK_s)^{\delta}),\ g_2^{\delta}) \\ &= (\hat{e}(w, g_2^{\alpha}) \cdot \hat{e}(g_1, g_2)^{y \cdot x \cdot \delta},\ g_2^{\delta}). \end{aligned}$$

7. $\mathsf{Test}_1(c_w, t_*, t_w, SK_s)$: The algorithm tests whether the ciphertext contains the same massage as the trapdoor. The algorithm parses c_w as (c_1, c_2), t_* as (t_1, t_2, t_3, t_4), t_w as (t_5, t_6) and defines:

$$t_7 = \frac{t_1}{t_2^x},\ t_8 = \frac{t_3}{t_4^x}, \tilde{a} = \frac{\hat{e}(PK_r, t_6^x) \cdot \hat{e}(c_1, t_7)}{t_5}, \tilde{b} = \hat{e}(c_2, t_8).$$

   Finally, the algorithm checks whether $\tilde{a} = \tilde{b}$. If this equation holds, the algorithm outputs TRUE indicating that the ciphertext contains the same massage as the trapdoor, otherwise it outputs FALSE.
8. $\mathsf{Test}_2(c_w, t_*, w, SK_s)$: The algorithm tests whether the ciphertext contains the word w. The algorithm parses c_w as (c_1, c_2), t_* as (t_1, t_2, t_3, t_4), and defines:

$$t_7 = \frac{t_1}{t_2^x}, t_8 = \frac{t_3}{t_4^x}, \tilde{c} = \hat{e}(c_1, t_7), \hat{d} = \hat{e}(c_2, t_8).$$

   Finally, the algorithm checks whether $\frac{\tilde{c}}{\hat{d}} = \hat{e}(w, t_7)$. If this equation holds, the algorithm outputs TRUE indicating that ciphertext contains the word w, otherwise it outputs FALSE.
9. $\mathsf{Decrypt}(SK_r, c_w)$: The algorithm outputs:

$$w = \frac{c_1}{c_2^y}.$$

As indicated by the above scheme, in the PKEDS formulation, both Test_1 and Test_2 require a master trapdoor t_* as input. Since the master trapdoor allows a server to test any word at its own choice, therefore, only one-wayness property has been defined against servers in [23], even if a server only performs searches based on message-dependent trapdoors (namely, only executing Test_1).

We argue that this is an undesirable feature. Referring to the scenario described in Section 1, different servers may be involved in practice: one server performs virus scanning by running Test_2, while the other server supports search and email retrieval service by running Test_1. The rationale behind this is that users often forward their emails from different accounts (e.g. those from previous employers) to an email archive (e.g. Gmail) and retrieve emails from the archive afterwards. In such a situation, it is unnecessary to assign a master trapdoor to the servers, which only perform search based on message-dependent trapdoors. Correspondingly, better security than one-wayness should be achieved against such servers.

As to the efficiency of this scheme, the Test_1 algorithm requires 3 exponentiations and 3 pairings while the Test_2 algorithm requires 2 exponentiations and 3 pairings. We note that, in both algorithms, t_7 and t_8 can be pre-computed and used in all algorithm executions. By doing so, two exponentiations will be saved. In addition, in the $\mathsf{Delegate}$ algorithm, the value of $g_2^{y \cdot \alpha}$ needs to be computed during each algorithm execution. In fact this can be avoided by defining the receiver's key generation algorithm as follows.

– $\mathsf{rKeyGen}(pp)$: Run by a receiver, this algorithm selects $y \in_R \mathbb{Z}_p$ and $x \in \mathbb{G}_2$, and sets $g_2^{\alpha} = x$. It outputs the receiver's private/public key pair:

$$(SK_r, PK_r) = ((y, (g_2^{\alpha})^y),\ g_1^y).$$

In the security model described in [23], the authors considered the multiple receivers. In fact, this is unnecessary because any attacker can trivially act as a valid receiver without any interaction with other entities. This is similar to the situation to public key encryption, where the attacker can generate key pairs for itself but we do not need to consider this explicitly in the (security) formulation. On the other hand, it is more likely there are multiple servers and a receiver may let all of them to search over his encrypted data. However, the security model in [23] does not consider this case. In the process of proving the property of ciphertext indistinguishability (CI-ATK) in [23], the authors publish the server's key pair (PK_s, SK_s) but prevent the attacker from obtaining the server's master trapdoor $t_{*,s}$. This is conflict with their security model, in which the attacker can query the server's master trapdoor $t_{*,s}$. This makes the proof invalid. Nevertheless, we do not claim that the scheme is insecure in their security model. In the proof of trapdoor indistinguishability, the authors let the simulator pick a random α to construct the trapdoor. However, this value should equal to the α value chosen in the key generation phase.

3 Refined Definition and Security Model

In this section, we present a refined formulation for PKEDS and propose a new security model.

3.1 Refined Definition for PKEDS

In the PKEDS setting, there are the following types of entities:

- A receiver, which is supposed to receive encrypted messages.
- The servers, which receive message-dependent tokens and/or master tokens from the receiver. Based on a message-dependent token, a server can test whether the message encoded in a token is equal to that in any given ciphertext. Based on a master trapdoor, a server can choose any target message and test whether it is equal to that in any given ciphertext.
- Message senders, which send messages to the receiver. Note that in the public key setting, every entity (including the servers) can be a message sender.

Formally, a PKEDS scheme consists of the following nine algorithms, where (rKeyGen, Encrypt, Decrypt) define a standard PKE scheme.

- Setup(λ): This algorithm takes as input a security parameter λ and outputs public parameters pp. In practice, these parameters may be standardized, and accepted by all entities.
- rKeyGen(pp): Run by the receiver, this algorithm outputs a public/private key pair (PK_r, SK_r). Let the message space be $\mathcal{W}$.
- sKeyGen(pp): Run by a server, this algorithm outputs a public/private key pair (PK_s, SK_s).
- Encrypt(w, PK_r): Run by a message sender, this algorithm outputs a ciphertext c_w for a message w.
- Delegate(PK_s, SK_r): Run by the receiver with private key SK_r, this algorithm outputs a master token $t_{*,s}$ for the server with public key PK_s.
- TrapGen(w, PK_s, SK_r): Run by the receiver with private key SK_r, this algorithm generates a token $t_{w,s}$ for the server with public key PK_s.
- Test$_1$($c_w, t_{w,s}, SK_s$): Run by the server with private key SK_s and trapdoor t_w, the algorithm returns 1 if w in t_w equals to the plaintext encrypted in c_w and 0 otherwise. After running the algorithm, the server should securely delete the trapdoor $t_{w,s}$.

– $\mathsf{Test}_2(c_w, w, t_{*,s}, SK_s)$: Run by the server with the master trapdoor $t_{*,s}$ and private key SK_s, the algorithm returns 1 if w is encrypted in c and 0 otherwise, for any w. For security reasons, the server should securely store $t_{*,s}$ as for its private key SK_s. Or, the server can securely delete $t_{*,s}$ and ask the receiver to re-send it when necessary.
– $\mathsf{Decrypt}(c, SK_r)$: Run by the receiver with private key SK_r, this algorithm outputs a plaintext w or an error message $\perp$.

As in the case of PKE schemes, we assume that the message senders possess valid copies of the public keys of the receiver and the receiver possesses valid copies of the public keys of the servers. How to securely distribute these public keys should follow some standard practice, and we skip the discussion in this paper. Besides this trusted setup assumption, the communications between parties are carried out in an open network, where an attacker can eavesdrop on the exchanged messages.

Corresponding to the functionality descriptions of various participants in the system, we consider the following types of attacker:

– Malicious outside attacker: This type of attacker does not have access to any type of tokens, and essentially has the same privilege as an IND-CPA attacker for PKE schemes.
– Honest-but-curious server(s): This type of attacker may have been assigned both message-dependent tokens and master tokens.

As to security, we assume that the attacker's purpose is to learn the information in ciphertexts or trapdoors. In practice, an attacker may simply mount a DoS (denial of service) by manipulating (or, deleting) the communication messages. How to prevent such attacks is an open problem not only for PKEDS schemes but also for other schemes, and will not be addressed in this paper.

3.2 Definitions of Various Properties

Similar to the case for other primitives, the first property we want is soundness, defined as follows.

Definition 1. *A PKEDS scheme is sound if, for* $(PK_r, SK_r) = \mathsf{rKeyGen}(pp)$ *and* $(PK_s, SK_s) = \mathsf{sKeyGen}(pp)$*, the following conditions are satisfied.*

1. *For any* $w \in \mathcal{W}$*, the probability that the equality* $\mathsf{Decrypt}(\mathsf{Encrypt}(w, PK_r), SK_r) = w$ *does not hold is negligible.*

2. *For any w, the probability that the equalities* $\mathsf{Test}_1(\mathsf{Encrypt}(w, PK_r), t_{w,s}, SK_s) = 1$ *and* $\mathsf{Test}_2(\mathsf{Encrypt}(w, PK_r), w, t_{*,s}, SK_s) = 1$ *do not hold is negligible.*
3. *For any c, if* $\mathsf{Test}_1(c, t_{w,s}, SK_s) = 1$ *or* $\mathsf{Test}_2(c, w, t_{*,s}, SK_s) = 1$*, then the probability that* $\mathsf{Decrypt}(c, SK_r) = w$ *does not hold is negligible.*

The first condition basically guarantees that the encryption/decryption functionality works well, and the second condition guarantees that the test algorithms works well for honestly generated ciphertexts. The third condition eliminates trivial solutions, where the test algorithms always output 1.

Ciphertext indistinguishability against servers with message-dependent trapdoors. This property is an adaption of the security definition of PEKS [6] to PKEDS. It says that, given some ciphertexts and some message-dependent trapdoors, an attacker cannot learn anything about the plaintexts.

Definition 2. *A PKEDS scheme achieves ciphertext indistinguishability against servers with message-dependent trapdoors, if the attacker's advantage (i.e. $|Pr[b' = b] - \frac{1}{2}|$) is negligible in the attack game depicted in Fig 1.*

1. Setup: The challenger runs the $\mathsf{rKeyGen}$ algorithm to generate a public/private key pair(PK_r, SK_r), and publishes PK_r.
2. Phase 1: The attacker is allowed to issue the following types of oracle queries:
   - $\mathsf{sKeyGen}$ query: The challenger runs the KenGen algorithm to generate a public/private key pair(PK_s, SK_s) and returns PK_s.
   - $\mathsf{sKeyGen}^+$ query: The challenger runs the KenGen algorithm to generate a public/private key pair(PK_s, SK_s) and returns (PK_s, SK_s).
   - $\mathsf{Delegate}$ query with a public key PK_s: The challenger returns $\mathsf{Delegate}(PK_s, SK_r)$.
   - $\mathsf{TrapGen}$ query with a message w and PK_s as input: The challenger returns $\mathsf{TrapGen}(w, PK_s, SK_r)$.

   At some point, the attacker sends two messages w_0, w_1 to the challenger for a challenge. For any PK_{s^*} which is the output of an $\mathsf{sKeyGen}^+$ oracle query, there are two restrictions: (1) neither w_0 nor w_1 should have been queried to the $\mathsf{TrapGen}$ oracle with PK_{s^*}; (2) the $\mathsf{Delegate}$ oracle should not have been queried with PK_{s^*}.
3. Challenge: The challenger selects $b \in_R \{0, 1\}$ and sends $c_{w_b} = \mathsf{Encrypt}(w_b, PK_r)$ to the attacker.
4. Phase 2: The attacker is allowed to issue the same types of oracle queries as in Phase 1. At some point, the attacker terminates by outputting a guess $b' \in \{0, 1\}$.

Fig. 1. Security against servers with Message-dependent Trapdoors

Ciphertext One-wayness against servers with both types of trapdoors. This property means that, it is hard for an attacker to invert any ciphertext and to

learn the word even if it holds all server's private key, the master trapdoor and the message-dependent trapdoor associated with that word. This property is defined in Definition 3.

Definition 3. *A PKEDS scheme achieves ciphertext one-wayness against the servers, if the attacker's advantage (i.e. $|Pr[w' = w]|$) is negligible in the attack game depicted in Fig 2.*

1. Setup: The challenger runs the rKeyGen algorithm to generate a public/private key pair(PK_r, SK_r), and publishes PK_r.
2. Phase 1: The attacker is allowed to issue the following types of oracle queries: sKeyGen, sKeyGen$^+$, Delegate, and TrapGen. The queries are answered in the same way as in Fig. 1. At some point, the attacker asks the challenger for a challenge with respect to some PK_s.
3. Challenge: The challenger selects $w^* \in_R \mathcal{W}$ and sends the challenge ciphertext $c_{w^*} =$ Encrypt(w^*, PK_r) and the challenge trapdoor TrapGen(w^*, PK_s, SK_r) to the attacker.
4. Phase 2: The attacker is allowed to issue the same types of oracle queries as in Phase 1. At some point, the attacker terminates by outputting a word $w' \in \mathcal{W}$.

Fig. 2. Ciphertext One-wayness against all Servers

Trapdoor indistinguishability. This property says that an attacker cannot learn anything about the plaintexts encoded in message-dependent trapdoors sent to a server, where the attacker can be an outsider or other servers. In practice, this prevents the attacker from knowing what the receiver has searching for. In the attack game, depicted in Fig. 3, we allow the attacker to know the receiver's private key. It captures our intention that, even if the receiver's private key is leaked or compromised, then the attacker still cannot figure out what the receiver has searched for. This is similar to the forward security property in key establishment protocols. Formally, this property is defined in Definition 4.

Definition 4. *A PKEDS scheme achieves trapdoor indistinguishability against an outside attacker, if the attacker's advantage (i.e. $|Pr[b' = b] - \frac{1}{2}|$) is negligible in the attack game depicted in Fig 3.*

Note that this definition is different from that proposed in [23], because we assume a multi-server setting.

1. Setup: The challenger runs the rKeyGen algorithm to generate a public/private key pair(PK_r, SK_r), and publishes PK_r.
2. Phase 1: The attacker is allowed to issue the following types of oracle queries: sKeyGen, sKeyGen$^+$, Delegate, and TrapGen. The queries are answered in the same way as in Fig. 1. At some point, the attacker sends two messages w_0, w_1 and PK_{s^*} to the challenger for a challenge. It is required that PK_{s^*} is the output of an sKeyGen oracle query, namely SK_{s^*} is not accessible to the attacker.
3. Challenge: The challenger selects $b \in_R \{0, 1\}$ and sends $t_{w_b} = \mathsf{TrapGen}(w_b, PK_{s^*}, SK_r)$ to the attacker.
4. Phase 2: The attacker is allowed to issue the same types of oracle queries as in Phase 1. In addition, the attacker can request SK_r. At some point, the attacker terminates by outputting a guess $b' \in \{0, 1\}$.

Fig. 3. Trapdoor Indistinguishability

4 A New PKEDS Scheme

In this section, we first propose a new scheme and then prove its security in our security model. Our construction makes use of bilinear groups. Let $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$ be groups of prime order p, and let g_1 and g_2 be generator of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. A bilinear map $\hat{e}$: $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ has the following properties[7]:

1. Bilinearity: for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p^*$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
2. Non-degeneracy: $\hat{e}(g_1, g_2) \neq 1$.
3. Efficient computability: There is a polynomial time algorithm to compute $\hat{e}(u, v)$, for any $u \in \mathbb{G}_1$ and $v \in \mathbb{G}_2$.

Similar to the construction in [23], we employ ElGamal in bilinear groups as the main building block in the proposed scheme and use the bilinear property to perform searches. Moreover, we employ the Boneh-Franklin IBE scheme[7] to provide additional components in the Encrypt and TrapGen algorithms. The *key privacy property* of Boneh-Franklin IBE scheme allows us to prove the ciphertext indistinguishability against servers with message-dependent trapdoors.

4.1 Description of the Proposed Scheme

The algorithms of the proposed PKEDS scheme are defined as follows.

1. Setup(λ): On input of the security parameter λ, the algorithm outputs public parameters (pp) which contain the description of groups $\mathbb{G}_1, \mathbb{G}_2$ of order p, the bilinear map $\hat{e}$: $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, generators g_1 and g_2 of groups $\mathbb{G}_1$ and

$\mathbb{G}_2$ respectively. Additionally, let $(\mathsf{Enc}, \mathsf{Dec})$ be a symmetric key encryption scheme, and KDF be a key derivation function.

2. $\mathsf{sKeyGen}(pp)$: On input of the public parameters pp, the algorithm picks uniformly at random $x \in_R \mathbb{Z}_p$ and outputs a key pair:

$$(SK_s, PK_s) = (x, g_2^x)$$

3. $\mathsf{rKeyGen}(pp)$: On input of the public parameters pp, the algorithm picks uniformly at random $y \in_R \mathbb{Z}_p$ and $h \in_R \mathbb{G}_2$, and outputs the receiver's key pair:

$$(SK_r, PK_r) = ((y, h^y, MSK^\dagger), (g_1^y, h, MPK^\dagger)),$$

where $(MPK^\dagger, MSK^\dagger)$ is the master puclic/private key pair for the Boneh-Franklin IBE scheme [7].

4. $\mathsf{Encrypt}(w, PK_r)$: On input of a message $w \in \mathbb{G}_1$ and the receiver's public key PK_r, the algorithm picks uniformly at random $k \in_R \mathbb{Z}_p$ and outputs a ciphertext:

$$\begin{aligned} c_w &= (c_1, c_2, c_3) \\ &= (w \cdot g_1^{y \cdot k}, g_1^k, \mathsf{Encrypt}^\dagger(g_2^k, w)). \end{aligned}$$

The value $\mathsf{Encrypt}^\dagger(g_2^k, w)$ represents a ciphertext of g_2^k under the identity w in the Boneh-Franklin IBE scheme[7].

5. $\mathsf{TrapGen}(w, PK_s, SK_r)$: On input of a message w, a server's public key $PK_s = g_2^x$, and the receiver's private key SK_r, the algorithm performs as follows.
   (a) Compute $t_3 = \hat{e}(w, g_2)$ and $SK_w^\dagger$.
   (b) Select uniformly at random $r_2 \in_R \mathbb{G}_2$ and $r_3 \in_R \mathbb{Z}_p$, and compute the following:

$$t_4 = r_2 \cdot g_2^{x \cdot r_3},\ t_5 = g_2^{r_3},\ t_6 = \mathsf{Enc}(SK_w^\dagger || t_3, \mathsf{KDF}(r_2)).$$

   (c) Output the trapdoor $t_{w,s} = (t_4, t_5, t_6)$.

   The value $SK_w^\dagger$ represents the secret key corresponding to the identity w in the Boneh-Franklin IBE scheme[7].

6. $\mathsf{Test}_1(c_w, t_{w,s}, SK_s)$: On input of a ciphertext c_w, a message-dependent trapdoor $t_{w,s}$ and a server's private key SK_s, the algorithm performs as follows.
   (a) Parse c_w as (c_1, c_2, c_3) and $t_{w,s}$ as (t_4, t_5, t_6).
   (b) Compute r_2 from t_4, t_5 by an ElGamal decryption, and decrypt t_6 with $\mathsf{KDF}(r_2)$ to obtain $(SK_w^\dagger, t_3)$.
   (c) Run the decryption algorithm of the Boneh-Franklin scheme to decrypt c_3 using $SK_w^\dagger$, and recover g_2^k.

(d) Compute $\tilde{a}$ as follows:

$$\tilde{a} = \frac{\hat{e}(c_1, g_2)}{\hat{e}(g_1^y, g_2^k)}.$$

(e) Check whether the following two equalities hold. If both of them hold, the algorithm outputs 1, otherwise outputs 0.

$$\tilde{a} = t_3,\ \hat{e}(g_1, g_2^k) = \hat{e}(g_1^k, g_2).$$

7. $\mathsf{Delegate}(PK_s, SK_r)$: On input of a server's public key PK_s and the receiver's private key SK_r, the algorithm picks uniformly at random $r_1 \in_R \mathbb{Z}_p$ and outputs the master trapdoor $t_{*,s}$,

$$t_{*,s} = (t_1, t_2) = (h^y \cdot g_2^{x \cdot r_1}, g_2^{r_1})$$

After receiving $t_{*,s}$, the server with private key SK_s can recover $h^y = \frac{t_1}{t_2^x}$

8. $\mathsf{Test}_2(c_w, w, t_{*,s}, SK_s)$: On input of a ciphertext c_w, a message w, a master trapdoor $t_{*,s}$ and a server's private key SK_s, the algorithm performs as follows.
   (a) Parse c_w as (c_1, c_2, c_3).
   (b) Compute $\tilde{c}$ and $\tilde{d}$ as follows:

$$\tilde{c} = \hat{e}(\frac{c_1}{w}, h),\ \tilde{d} = \hat{e}(c_2, h^y).$$

   Note that h^y can be pre-computed by the server once receiving $t_{*,s}$.
   (c) Check whether $\tilde{c} = \tilde{d}$. If this equation holds, the algorithm outputs 1, otherwise outputs 0.

9. $\mathsf{Decrypt}(c_w, SK_r)$: On input of the ciphertext c_w and the receiver's private key SK_r, the algorithm outputs $w = \frac{c_1}{c_2^y}$.

Before going ahead, we show that the above scheme is sound.

Theorem 1. *The proposed scheme is sound under Definition 1.*

Proof. The first and the second requirements are straightforward based on the definitions of the encryption and test algorithms.

As to the third requirement, we prove two things. Firstly, given a ciphertext $c = (c_1, c_2, c_3) = (w' \cdot g_1^{y \cdot k}, g_1^k, \mathsf{Encrypt}^{\dagger}(g_2^{k'}, w))$, suppose that $\mathsf{Test}_1(c, t_{w,s}, SK_s) = 1$. Based on the definition of Test_1, we have the following equalities.

$$\frac{\hat{e}(c_1, g_2)}{\hat{e}(g_1^y, g_2^{k'})} = \hat{e}(w, g_2),\ \hat{e}(g_1, g_2^{k'}) = \hat{e}(g_1^k, g_2).$$

They imply $k' = k$ and $w' = w$. As a result, $\mathsf{Decrypt}(c, SK_r) = w$. Secondly, given a ciphertext $c = (c_1, c_2, c_3) = (w' \cdot g_1^{y \cdot k}, g_1^k, \mathsf{Encrypt}^{\dagger}(g_2^{k'}, w))$, suppose that $\mathsf{Test}_2(c, w, t_{*,s}, SK_s) = 1$. Based on the definition of Test_2, we have $\hat{e}(\frac{c_1}{w}, h) = \hat{e}(c_2, h^y)$ which implies $w' = w$. As a result, $\mathsf{Decrypt}(c, SK_r) = w$. In summary, the third requirement is satisfied. □

4.2 Efficiency Comparison with the INHJ Scheme

In the following table, we compare the number of calculations from the view of Sender, Server(s), and Receiver between our scheme and PKEDS. It is clear that our scheme is much more efficient than the INHJ scheme, in particular from the perspective of server(s).

		INHJ Scheme	Our Scheme
Server(s)	sKeyGen	1 Exp	1 Exp
	Test_1	3 Exp + 3 Pairing	1 Exp + 5 Pairing
	Test_2	2 Exp + 3 Pairing	2 Pairing
Receiver	rKeyGen	2 Exp	3 Exp
	Delegate	5 Exp	2 Exp
	TrapGen	2 Exp + 2 Pairing	3 Exp + 1 Pairing
	Decrypt	1 Exp	1 Exp
Senders	Encrypt	2 Exp	4 Exp + 1 Pairing

Computational Complexity Comparison

4.3 Security Analysis

We make the Decisional Diffie-Hellman (DDH) assumption in both $\mathbb{G}_1$ and $\mathbb{G}_2$ [4]. This means that there is no known efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$

Definition 5. *Decisional Diffie-Hellman Assumption in $\mathbb{G}_i$ ($i = 1$ or $i = 2$): Given a tuple $(g_i, g_i^a, g_i^b, g_i^c) \in \mathbb{G}_i^4$, where g_i is a generator of $\mathbb{G}_i$ and $a, b \in_R \mathbb{Z}_p$, decide whether $c = ab$ or $c \in_R \mathbb{Z}_p$.*

Next, we introduce a modified version of Computational Diffie-Hellman assumption in pairing setting, following [23]. Note that this assumption is weaker than the BDH-3 assumption [13].

Definition 6. *Modified Computational Diffie-Hellman (mCDH) Assumption: Given tuples* $(g_1, g_1^a, g_1^b) \in \mathbb{G}_1$ *and* $(g_2, g_2^a) \in \mathbb{G}_2$ *where* $a, b \in_R \mathbb{Z}_p$*, it is hard to compute* g_1^{ab}*.*

Next, we prove that the proposed PKEDS scheme achieve the three properties under Definitions 2, 3, and 4.

As said before, in order to prove the property against servers with message-dependent trapdoors, we have introduced the Boneh-Franklin scheme [8] in our construction. Boneh and Franklin proved the semantic security based on the standard BDH (bilinear Diffie-Hellman) assumption, and Holt proved the key privacy property based on the same assumption [20]. Informally, the key privacy property guarantees that an attacker is not able to distinguish the encryptions of w based on the two identities id_0 and id_1, even if it is allowed to choose (w, id_0, id_1) and query the secret keys with respect to all identities except for id_0 and id_1. A formal definition and proof can be found in [20].

Theorem 2. *The proposed scheme achieves ciphertext indistinguishability under Definition 2, based on the following assumptions: (1) DDH assumption in both* $\mathbb{G}_1$ *and* $\mathbb{G}_2$*; (2)* (Enc, Dec) *is IND-CPA secure; (3) Boneh-Franklin IBE scheme achieves semantic security and key privacy property [8, 20].*

Proof. Suppose that an outside adversary $\mathcal{A}$ has advantage ϵ in the attack game shown in Fig. 1. Denote this game as Game0.

Next, we consider a new game *Game*1, in which the challenger performs in the same way as in Game0, except the following.

– For a Delegate query with input PK_s, if PK_s is the output of a sKeyGen query, the challenger picks uniformly at random $r_1 \in_R \mathbb{Z}_p$ and outputs the master trapdoor $t_{*,s}$, where $R \in_R \mathbb{G}_2$ and

$$t_{*,s} = (t_1, t_2) = (R \cdot g_2^{x \cdot r_1}, g_2^{r_1})$$

Suppose that the attacker has advantage ϵ_1 in this game. Note that, in the attack game, the Delegate oracle is not allowed to be queried with PK_{s^*} for which the attacker knows SK_{s^*}. Based on the semantic security of ElGamal encryption, then $|\epsilon_1 - \epsilon|$ is negligible based on the DDH assumption in $\mathbb{G}_2$.

Next, we consider a new game *Game*2, in which the challenger performs in the same way as in *Game*1, except that the challenge is generated as follows.

1. Select uniformly at random $b, b^* \in_R \{0, 1\}$.

2. Select uniformly at random $k \in_R \mathbb{Z}_p$, and compute the following:

$$c_1 = w_b \cdot g_1^{y \cdot k},\ c_2 = g_1^k,\ c_3 = \mathsf{Encrypt}^{\dagger}(g_2^k, w_{b^*}).$$

3. Output the ciphertext $c_{w_b} = (c_1, c_2, c_3)$.

Suppose that the attacker has advantage ϵ_2 in this game. Note that, in the attack game, neither w_0 nor w_1 is allowed to be queried to the $\mathsf{TrapGen}$ oracle with PK_{s^*}, which is the output of an $\mathsf{sKeyGen}^{+}$ oracle query. This restriction means that the Boneh-Franklin secret keys corresponding to w_0 and w_1 are not accessible to the attacker. As a result, based on the key privacy property of the Boneh-Franklin IBE scheme [20], $|\Pr[b' = b|b^* = b] - \Pr[b' = b|b^* \neq b]|$ is negligible (if this is not the case, then we can straightforwardly build an attacker to break the IND-II-CPA security (e.g. key privacy) proved in [20]). As a result, $|\epsilon_2 - \epsilon_1|$ is negligible based on the key privacy property of Boneh-Franklin scheme [8, 20].

Next, we consider a new game *Game*3, in which the challenger performs in the same way as in *Game*2, except that the challenge is generated as follows.

1. Select uniformly at random $b, b^* \in_R \{0, 1\}$.
2. Select uniformly at random $k \in_R \mathbb{Z}_p$ and $R \in_R \mathbb{G}_2$, and compute the following:

$$c_1 = w_b \cdot g_1^{y \cdot k},\ c_2 = g_1^k,\ c_3 = \mathsf{Encrypt}^{\dagger}(R, w_{b^*}).$$

3. Output the ciphertext $c_{w_b} = (c_1, c_2, c_3)$.

Suppose that the attacker has advantage ϵ_3 in this game. As stated above, the Boneh-Franklin secret keys corresponding to w_0 and w_1 are not accessible to the attacker. As a result, based on the semantic security of Boneh-Franklin scheme [8], $|\epsilon_3 - \epsilon_2|$ is negligible .

In *Game*3, the challenge is a standard ElGamal challenge (c_3 is an encryption of random message), therefore ϵ_3 is negligible based on the DDH assumption in $\mathbb{G}_1$. Based on the above analysis, ϵ is also negligible based on all the assumptions mentioned in the theorem. □

Theorem 3. *The proposed scheme achieves ciphertext one-wayness against the servers under Definition 3, based on the mCDH assumption.*

Proof. Referring to Definition 3, the attacker's queries can be faithfully answered given the following information: $(g_1, h, g_1^y, h^y, MSK^{\dagger})$ together with the challenge $(w \cdot g_1^{y \cdot k}, g_1^k)$. The theorem immediately follows from the mCDH assumption. □

Theorem 4. *The proposed scheme achieves trapdoor indistinguishability under Definition 4, based on the following assumptions: (1) DDH assumption in* $\mathbb{G}_2$*; (2)* (Enc, Dec) *is IND-CPA secure; (3)* KDF *is modeled as a random oracle.*

Proof. Suppose that an outside adversary $\mathcal{A}$ has advantage ϵ in the attack game shown in Fig. 3. Denote this game as Game0. Let the challenged key pair be denoted as $(SK_{s^*}, PK_{s^*}) = (x^*, g_2^{x^*})$. Note that the challenge is generated by the challenger as follows.

1. Select uniformly at random $b \in_R \{0, 1\}$.
2. Compute $t_3 = \hat{e}(w_b, g_2)$.
3. Select uniformly at random $r_2 \in_R \mathbb{G}_2$ and $r_3 \in_R \mathbb{Z}_p$, and compute the following:
$$t_4 = r_2 \cdot g_2^{x^* \cdot r_3},\ t_5 = g_2^{r_3},\ t_6 = \mathsf{Enc}(SK_w^{\dagger} \| t_3, \mathsf{KDF}(r_2)).$$
4. Output the trapdoor $t_{w_b, s^*} = (t_4, t_5, t_6)$.

Next, we consider a new game *Game*1, in which the challenger performs in the same way as in Game1, except for the challenge generation.

– The challenger performs as follows.
  1. Select uniformly at random $r_2 \in_R \mathbb{G}_2$, $r_3 \in_R \mathbb{Z}_p$, $r_4 \in_R \mathbb{G}_T$, and r_4 is a randomly chosen group element in Boneh-Franklin IBE, and compute the following:
  $$t_4 = r_2 \cdot g_2^{x^* \cdot r_3},\ t_5 = g_2^{r_3},\ t_6 = \mathsf{Enc}(r_4 \| r_3, K),$$
  where K is a randomly chosen key from the key space of (Enc, Dec).
  2. Output the trapdoor $t_{w_b, s} = (t_4, t_5, t_6)$.

Suppose that the attacker has advantage ϵ_1 in this game. Note that, in this attack game, the attacker is not allowed to issue any decryption query with respect to (SK_{s^*}, PK_{s^*}). If KDF is modeled as a random oracle, then $|\epsilon_1 - \epsilon|$ is negligible based on the CDH assumption in $\mathbb{G}_2$ and the IND-CPA security of (Enc, Dec).

In *Game*1, $\epsilon_1 = 0$ since the challenge is independent from b. Based on the above analysis, ϵ is also negligible based on all the assumptions mentioned in the theorem. □

5 Related Work

The notion of searches on encrypted data was first introduced by Song et al. [31]. In [31], the authors present a cryptographic scheme for the problem of searching on encrypted data and provide proofs of security for the resulting crypto systems. Since then, many extensions and revisions have appeared. Keyword search is one of these extensions. The Public Key Encryption with Keyword Search(PEKS) scheme was first proposed by Boneh et al. [6]. The proposed scheme is based on anonymous identity-based encryption (IBE) which is introduced in [7]. Their scheme can be applied in store-and-forward system, such as an email system which a user can send trapdoors to the mail server and the sever will identify messages which contain some specific keywords. This construction was later improved by several researchers.

Abdalla et al. [1] discussed a consistency flaw in [6] and provided a transform of an anonymous IBE scheme to a secure PEKS scheme that guarantees consistency. Crescenzo et al. [14] proposed a PEKS construction based on Jacobi symbol and Khader et al. [25] introduced how to construct PEKS based on K-Resilient IBE. Boyen et al. [10] presented an identity-based cryptosystem that features fully anonymous ciphertexts and hierarchical key delegation. In addition, they gave a proof of security in the standard model, based on the mild Decision Linear complexity assumption in bilinear groups. Golle et al. [18] first proposed the notion of Conjunctive Keywords Searchable Encryption in private key model of keyword search. The notion of Public Key Encryption with Conjunctive field Keyword Search was first introduced by Park et al. [26]. Later, Byun et al. [11] and Hwang et al. [22] improved the efficiency of the conjunctive keyword search, but neither of them supported subset keyword search. Boneh et al. [9] constructed public-key systems which supported comparison queries on encrypted data as well as more general queries such as subset queries. However, its efficiency is not satisfying. Zhang et al. [34] discussed the flaws of the scheme proposed in [9] and gave out a more efficient construction of Public Key Encryption with Conjunctive-Subset Keywords Search(PECSK)scheme.

Boneh's scheme [6] was based on the unrealistic assumption of a secure channel between the receiver and the server. Baek et al. [3] proposed an efficient PEKS scheme which can remove a secure channel. Fang et al. [15] proposed an efficient scheme without using any secure channels and its security does not use random oracles. The papers [12] [33] [29] [24] studied the off-line keyword guessing attacks on PEKS. In fact, all the schemes in [22] [3] [1] [2] [14] [25] [35] suffer from the keyword-guessing attack. To cope with this problem, Tang et al. [32] proposed a new concept named Public-key Encryption with Registered Keyword Search(PERKS), which is immune to the off-line keyword guessing attack. Rhee et al. [28] introduced the concept of "trapdoor indistinguishability" and showed that this property was a sufficient condition for thwarting keyword-guessing attacks.

However, the PEKS scheme does not allow the receiver to decrypt the keywords. In other words, the encryption of the keywords is irreversible. This property of PEKS scheme limits its application in some situations. Beak et al. [2] proposed the first scheme which combines a Public Key Encryption (PKE) scheme and a Public key Encryption with Keyword Search (PEKS) scheme, and showed that it was IND-PKE/PEKS-CCA secure in the random oracle model. Later, Zhang et al. [35] first defined a new security model and gave a generic construction which is secure under the new security model without random oracles. Unfortunately, both the schemes proposed in [2] and [35] do not allow to retrieve the PEKS keyword and do not guarantee any relation between messages and keywords. Fuhr et al. [16] proposed a construction for decryptable searchable encryption which makes use of one KEM, one IDKEM and a couple of hash functions in the random oracle model, but it does not allow to retrieve the PEKS messages as well. Hofheinz et al. [19] presented a searchable public key encryption with decryption (PEKSD) in the standard model. Their scheme is based on Rabins trapdoor one-way permutation[27], and they proved that the security of their scheme against adaptive chosen-ciphertext attacks (CCA security) is equivalent to the factoring assumption. Fang et al. [15] first constructed a PEKS scheme that allows to retrieve both the PEKS keywords and the messages without random oracle model, but it is not the scheme with a designated tester. Recently, Hu et al. [21] presented a secure decryptable searchable encryption with a designated tester. Roschke et al. [30] proposed a technique which detects malicious content in the encrypted data, but the technique in [30] uses the master secret key of the IBE to decrypt the ciphertext and then uses the virus scanner to scan the plaintext. Different from the scheme in [30], Ibraimi et al. [23] proposed a public-key encryption with delegated search scheme which can detect malicious content in the encrypted data without having to decrypt it. Their scheme used Type-3 pairings [17] which are employed by the server to search the ciphertext and the receiver to generate the trapdoor. The ciphertexts in the proposed scheme are both searchable and decryptable. They proved their security properties under the Symmetric External Diffie-Hellman (SXDH) assumption and the modified Computational Diffie-Hellman (mCDH) [13] assumption.

6 Conclusion

In this paper, we have revisited the concept of public key encryption with delegated keyword search (PKEDS), a concept initially proposed in [23]. We have shown that the existing formulation has some defects, the proposed scheme is unnecessarily inefficient and there are some flaws in the security proof. As a result, we have presented a refined formulation for PKEDS together with a new security model, and propose a new PKEDS scheme which is proven secure in the new security model. Nevertheless, there are some open research problems for this new primitive. In the proposed scheme, the encryption algorithm becomes

more complex than a standard ElGamal encryption, so does one of the test algorithms. It is interesting to investigate how to improve the efficiency of these algorithms. As to the security against servers with both types of trapdoors, we only consider one-wayness for the moment, it will be interesting to consider stronger security guarantees, such as achieving the notion proposed in [5].

References

1. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In V. Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference*, volume 3621 of *LNCS*, pages 205–222. Springer, 2005.
2. J. Baek, R. Safavi-Naini, and W. Susilo. On the integration of public key data encryption and public key encryption with keyword search. In S. K. Katsikas, J. Lopez, M. Backes, S. Gritzalis, and B. Preneel, editors, *Information Security, 9th International Conference, ISC 2006*, volume 4176 of *LNCS*, pages 217–232. Springer, 2006.
3. J. Baek, R. Safiavi-Naini, and W. Susilo. Public key encryption with keyword search revisited. In O. Gervasi, B. Murgante, A. Lagan, D. Taniar, Y. Mun, and M. Gavrilova, editors, *Computational Science and Its Applications C ICCSA 2008*, volume 5072 of *LNCS*, pages 1249–1259. Springer, 2008.
4. L. Ballard, M. Green, B. de Medeiros, and F. Monrose. Correlation-resistant storage via keyword-searchable encryption. Cryptology ePrint Archive: Report 2005/417, 2005.
5. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In *Advances in cryptology — CRYPTO 2007*, pages 535–552. Springer-Verlag, 2007.
6. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public Key Encryption with Keyword Search. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology — EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 506–522. Springer, 2004.
7. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In J. Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.
8. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In J. Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
9. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In S. P. Vadhan, editor, *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007*, volume 4392 of *LNCS*, pages 535–554. Springer, 2007.
10. X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In C. Dwork, editor, *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference*, volume 4117 of *LNCS*, pages 290–307. Springer, 2006.
11. J. W. Byun, D. H. Lee, and J. Lim. Efficient conjunctive keyword search on encrypted data storage system. In A. Atzeni and A. Lioy, editors, *Public Key Infrastructure, Third European PKI Workshop: Theory and Practice, EuroPKI 2006*, volume 4043 of *LNCS*, pages 184–196. Springer, 2006.

12. J. W. Byun, H. S. Rhee, H. Park, and D. H. Lee. Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In W. Jonker and M. Petkovic, editors, *Secure Data Management, Third VLDB Workshop, SDM 2006*, volume 4165 of *LNCS*, pages 75–83. Springer, 2006.
13. S. Chatterjee and A. Menezes. On cryptographic protocols employing asymmetric pairings - the role of psi revisited. *IACR Cryptology ePrint Archive 2009/480*, 2009.
14. G. Di Crescenzo and V. Saraswat. Public key encryption with searchable keywords based on Jacobi symbols. In K. Srinathan, C. P. Rangan, and M. Yung, editors, *Progress in Cryptology - INDOCRYPT 2007, 8th International Conference on Cryptology in India*, volume 4859 of *LNCS*, pages 282–296. Springer, 2007.
15. L. Fang, J. Wang, C. Ge, and Y. Ren. Decryptable public key encryption with keyword search schemes. *Journal of Digital Content Technology and its Applications*, 4(9):141–150, 2010.
16. T. Fuhr and P. Paillier. Decryptable searchable encryption. In W. Susilo, J. K. Liu, and Y. Mu, editors, *Provable Security, First International Conference, ProvSec 2007*, volume 4784 of *LNCS*, pages 228–236. Springer, 2007.
17. S. D. Galbraith, K. G. Paterson, and N. P. Smart. Pairings for cryptographers. *IACR Cryptology ePrint Archive*, 2006:165, 2006.
18. P. Golle, J. Staddon, and B. Waters. Secure conjunctive keyword search over encrypted data. In M. Jakobsson, M. Yung, and J. Zhou, editors, *Applied Cryptography and Network Security, Second International Conference, ACNS 2004*, volume 3089 of *LNCS*, pages 31–45. Springer, 2004.
19. D. Hofheinz and E. Kiltz. Practical chosen ciphertext secure encryption from factoring. In A. Joux, editor, *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, volume 5479 of *LNCS*, pages 313–332. Springer, 2009.
20. J. E. Holt. Key privacy for identity based encryption. Cryptology ePrint Archive: Report 2006/120, 2006.
21. C. Hu and P. Liu. Decryptable searchable encryption with a designated tester. *Procedia Engineering*, 15(0):1737–1741, 2011.
22. Y. H. Hwang and P. J. Lee. Public key encryption with conjunctive keyword search and its extension to a multi-user system. In T. Takagi, E. Okamoto, and T. Okamoto, editors, *Pairing-Based Cryptography - Pairing 2007, First International Conference*, volume 4575 of *LNCS*, pages 2–22. Springer, 2007.
23. L. Ibraimi, S. Nikova, P. H. Hartel, and W. Jonker. Public-key encryption with delegated search. In J. Lopez and G. Tsudik, editors, *Applied Cryptography and Network Security - 9th International Conference, ACNS 2011*, volume 6715 of *LNCS*, pages 532–549, 2011.
24. I. R. Jeong, J. O. Kwon, D. Hong, and D. H. Lee. Constructing peks schemes secure against keyword guessing attacks is possible? *Computer Communications*, 32(2):394–396, 2009.
25. D. Khader. Public key encryption with keyword search based on k-resilient ibe. In M. L. Gavrilova, O. Gervasi, V. Kumar, C. J. K. Tan, D. Taniar, A. Laganà, Y. Mun, and H. Choo, editors, *Computational Science and Its Applications - ICCSA 2006, International Conference*, volume 3982 of *LNCS*, pages 298–308. Springer, 2006.
26. D. J. Park, K. Kim, and P. J. Lee. Public key encryption with conjunctive field keyword search. In C. H. Lim and M. Yung, editors, *Information Security Applications, 5th International Workshop, WISA 2004*, volume 3325 of *LNCS*, pages 73–86. Springer, 2004.

27. M. O. Rabin. Digital signatures and public key functions as intractable as factorization. Technical report, MIT/LCS/TR-212, Massachusetts Institute of Technology, 1979.
28. H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee. Trapdoor security in a searchable public-key encryption scheme with a designated tester. *Journal of Systems and Software*, 83(5):763–771, 2010.
29. H. S. Rhee, W. Susilo, and H. Kim. Secure searchable public key encryption scheme against keyword guessing attacks. *IEICE Electronics Express*, 6(5):237–243, 2009.
30. S. Roschke, L. Ibraimi, F. Cheng, and C. Meinel. Secure communication using identity based encryption. In B. De Decker and I. Schaumüller-Bichl, editors, *Communications and Multimedia Security, 11th IFIP TC 6/TC 11 International Conference, CMS 2010*, volume 6109 of *LNCS*, pages 256–267. Springer, 2010.
31. D. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In *Proceedings of the 2000 IEEE symposiumon security and privacy*, pages 44–55. IEEE, May 2000.
32. Q. Tang and L. Chen. Public-key encryption with registered keyword search. In Fabio Martinelli and Bart Preneel, editors, *Proceeding of Public Key Infrastructure, 5th European PKI Workshop: Theory and Practice (EuroPKI 2009)*, volume 6391 of *LNCS*, pages 163–178. Springer, 2009.
33. W. Yau, S. Heng, and B. Goi. Off-line keyword guessing attacks on recent public key encryption with keyword search schemes. In C. Rong, M. G. Jaatun, F. E. Sandnes, L. T. Yang, and J. Ma, editors, *Autonomic and Trusted Computing, 5th International Conference, ATC 2008*, volume 5060 of *LNCS*, pages 100–105. Springer, 2008.
34. B. Zhang and F. Zhang. An efficient public key encryption with conjunctive-subset keywords search. *J. Network and Computer Applications*, 34(1):262–267, 2011.
35. R. Zhang and H. Imai. Generic combination of public key encryption with keyword search and public key encryption. In F. Bao, S. Ling, T. Okamoto, H. Wang, and C. Xing, editors, *Cryptology and Network Security, 6th International Conference, CANS 2007*, volume 4856 of *LNCS*, pages 159–174, 2007.