# An Investigation of Some Forward Security Properties for PEKS and IBE

Qiang Tang

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

**Abstract.** In cryptography, forward secrecy is a well-known property of key agreement protocols. It ensures that a session key remains secure even if one of the long-term secret keys is compromised in the future. In this paper, we investigate some forward security properties for Public-key Encryption with Keyword Search (PEKS) schemes, which allow a client to store encrypted data and delegate search operations to a server. The proposed properties guarantee that the client's privacy is protected to the maximum extent when his private key is compromised. Motivated by the generic transformation from anonymous Identity-Based Encryption (IBE) to PEKS, we correspondingly propose some forward security properties for IBE, in which case we assume the attacker learns the master secret key. We then study several existing PEKS and IBE schemes, including a PEKS scheme by Nishioka, an IBE scheme by Boneh, Raghunathan and Segev, and an IBE scheme by Arriaga, Tang and Ryan. Our analysis indicates that the proposed forward security properties can be achieved by some of these schemes if the attacker is RO-non-adaptive (the attacker does not define its distributions based on the random oracle). Finally, we show how to extend the Boyen-Waters anonymous IBE scheme to achieve the forward security properties for adaptive attackers.

## 1 Introduction

In the seminal work [8], Boneh et al. proposed the concept of Public-key Encryption with Keyword Search (PEKS) and formulated it as a cryptographic primitive with four algorithms ($\mathsf{KeyGen}$, $\mathsf{Encrypt}$, $\mathsf{TrapGen}$, $\mathsf{Test}$). PEKS is a two-party (i.e. client-server) primitive aiming at protecting a client, say Alice's, privacy in the following *encrypted email routing scenario*.

1. Alice runs a $\mathsf{KeyGen}$ algorithm to generate a public/private key pair $(PK, SK)$ and publishes $PK$.
2. When any user, say Bob, sends an email to Alice, he can generate a tag $\mathsf{Encrypt}(x, PK)$ for a keyword $x$ and attach it to the email (the email should be encrypted independently and the detail is omitted here). In the view of the email server, it has a list of emails indexed by $\mathsf{Encrypt}(x_1, PK)$, $\mathsf{Encrypt}(x_2, PK), \cdots$ respectively.
3. If Alice wants to retrieve those emails indexed with a keyword $y$, she sends a trapdoor $\mathsf{TrapGen}(y, SK)$ to the email server, which can then run an algorithm $\mathsf{Test}$ on the input $(\mathsf{TrapGen}(y, SK), \mathsf{Encrypt}(x_i, PK))$ for every $i \geq 1$ to figure out whether $y = x_i$.

As shown in this scenario, PEKS only supports equality testing (or, exact matching) of keywords. To support more complex search queries, a lot of extensions have been proposed. Among them, [13, 21, 22] support search queries with conjunctive keywords, [13, 27] support subset and range queries, and [23] supports disjunctions, polynomial equations, and inner products. In contrast to the large number of follow-up works to extend the PEKS functionality, very little has been done to investigate its full security capabilities and the only few we know are [3, 4, 9–11].

### 1.1 Problem Statement

With a PEKS scheme implemented, the server has a list of tags from message senders and a list of trapdoors from the client. As a result of the desired search functionality, the server can categorize the tags and the trapdoors into three groups.

- Group 1: the tags, which do not match any trapdoor.
- Group 2: the trapdoors, which do not match any tag.
- Group 3: the tags and trapdoors, which match at least one trapdoor or tag.

The seminal work [8] and most of follow-ups only considered the ciphertext privacy property, which captures the privacy of keywords encrypted by the tags in Group 1. Boneh et al. [10, 11] introduced a new function privacy property which captures privacy of keywords encrypted by the tags and trapdoors in Group 3. Arriaga, Tang, and Ryan [3, 4] introduced a new search pattern privacy property which captures the privacy of keywords in the trapdoors in Group 2. Moreover, they showed that these properties are not compatible with each other.

Ideally, a PEKS scheme should provide maximal protection for the keywords in all three groups. In this work, we try to answer two questions.

1. Are the security properties from [3, 4, 9–11] the strongest we can have?
2. How to construct PEKS schemes to achieve all properties simultaneously?

Interestingly, both questions can also be asked against the extensions of PEKS (and searchable encryption schemes in the symmetric-key setting as well). We leave this as a future work.

### 1.2 Concerning the Entropy of Keywords

With respect to PEKS and its extensions, there is a concern about the low entropy nature of keywords. For example, in the aforementioned email routing example, the entropy of keywords may not be very high. This makes people wonder the practicality of privacy properties for PKES, particularly the function privacy property [10, 11] and the search pattern privacy property [3, 4].

Nevertheless, we would like to argue that it makes a lot of sense to investigate the maximal level of security guarantees by PEKS. Theoretically, it is always interesting to study the strongest security properties for a cryptographic primitive. This has been done for many other primitives, such as encryption and signature schemes. Practically, it is not true that the keywords always have low entropy. When a PEKS scheme is deployed, the underlying application can always enrich the keyword set with some context information. For instance, instead of using the keyword "confidential", the application can use "confidential-project1457". A more effective way to augment the entropy of keywords is using pre-shared passwords between some message senders and the client. An extra advantage of this approach is that there is no need to store passwords given they are memorable. It is worth noting that augmenting the entropy of keywords may cause some efficiency issues (e.g. there may be several augmented keywords for the same keyword "confidential", so that the client needs to generate several trapdoors to search for all confidential emails). We regard this as a natural tradeoff between security and efficiency.

### 1.3 Our Contribution

Forward secrecy is a well-known property proposed for key agreement protocols [18, 20], and it ensures that a session key remains secure even if one of the long-term secret keys is compromised in the future. This concept has also been applied to other primitives, such as signature schemes [6]. Generally speaking, it is a valuable concept for all cryptographic primitives that involve a long-term secret key.

Firstly, we introduce two new forward-secure properties for PEKS. One is forward-secure function privacy, which aims at protecting the keywords in matched tags and trapdoors. The other is forward-secure trapdoor unlinkability, which aims at protecting the keywords in those trapdoors which do not match any tags. These two properties are augmented variants of those from [10, 11] and [3, 4] respectively. The augmentation lies not only in allowing the attacker to compromise the long-term secret key but also in giving it more flexibility to define the keyword distributions in the attack games. We then analyse a PEKS scheme by Nishioka [25] and show that it achieves our properties for RO-non-adaptive attackers which can not choose its distributions based on the random oracle.

Secondly, motivated by the fact that we can obtain a PEKS scheme from an anonymous IBE scheme through a generic transformation [1, 8], we introduce two new forward-secure properties for IBE by giving the master secret key the attacker in the attack games. It is worth noting that this is different from the forward security concept given in [30], where the focus is on the key evolution and the attacker is given secret keys corresponding to some identities. These properties directly lead to those forward-secure properties for PEKS as a result of the generic transformation. Similar to the case of PEKS, both properties are augmented variants of those from [10, 11] and [3, 4] respectively. We then analyse the $\mathcal{IBE}_{\mathsf{DLIN2}}$ scheme by Boneh, Raghunathan and Segev [10, 11], and show that it does not achieve the forward secure function privacy property. We also analyse an IBE scheme by Arriaga, Tang and Ryan [3, 4], and show that it achieves our properties for RO-non-adaptive attackers.

Thirdly, we introduce the concept of deterministic scrambler, which takes correlated random variables as input and outputs independent random variables. By pre-processing the identities with a deterministic scrambler scheme, an IBE scheme automatically achieves the forward-secure function privacy property against adaptive attackers. In contrast to the "extract-augment-combine" approach from [10, 11], we do not need to tweak the encryption and decryption algorithms of the underlying IBE scheme. We then extend the Boyen-Waters anonymous IBE scheme [14] with composite order bilinear groups and a deterministic scrambler scheme, and show that the resulted scheme achieves our forward-secure properties.

### 1.4 Organization

The rest of this paper is organized as follows. In Section 2, we present preliminaries on notation and hardness assumptions. In Section 3, we present an enhanced security model for PEKS with a focus on the forward security properties, and analyse the Nishioka scheme. In Section 4, we propose some forward security properties for IBE. In Section 5, we analyse two IBE schemes by Boneh, Raghunathan and Segev, and an IBE scheme by Arriaga, Tang and Ryan. In Section 6, we introduce the concept of deterministic scrambler and extend the Boyen-Waters Scheme. In Section 7, we present some remarks.

## 2 Preliminary

### 2.1 Notation

- $x\|y$ means the concatenation of $x$ and $y$, P.P.T. stands for probabilistic polynomial time.
- $x \overset{\$}{\leftarrow} \mathcal{A}^{O_1,O_2,\cdots}(m_1, m_2, \cdots)$ means that $x$ is the output of the algorithm $\mathcal{A}$ which runs with the input $m_1, m_2, \cdots$ and has access to oracles $O_1, O_2, \cdots$.
- When $X$ is a set, $x \overset{\$}{\leftarrow} X$ means that $x$ is chosen from $X$ uniformly at random, and $|X|$ means the size of $X$. When $\mathbb{D}$ is a distribution on the set $X$, $x \overset{\mathbb{D}}{\leftarrow} X$ means that $x$ is a value sampled from $X$ according to $\mathbb{D}$.
- We use bold letter, such as $\boldsymbol{X}$, to denote a vector or matrix. Given a vector $\boldsymbol{X}$, we use $\boldsymbol{X}^{(i)}$ to denote the $i$-th element in the vector. When $g$ is a group element, we use $g^{\boldsymbol{X}}$ to denote a new vector or matrix, whose elements are exponentiations of the corresponding elements in $\boldsymbol{X}$. For two vectors (or matrices) $\boldsymbol{Y}$ and $\boldsymbol{Z}$ whose elements are from a group, we use $\boldsymbol{Y} \otimes \boldsymbol{Z}$ to denote the new vector (or matrix) after pairwise group operations.
- A function $P(\lambda) : \mathbb{Z} \rightarrow \mathbb{R}$ is said to be negligible with respect to $\lambda$ if, for every polynomial $f(\lambda)$, there exists an integer $N_f$ such that $P(\lambda) < \frac{1}{f(\lambda)}$ for all $\lambda \geq N_f$. When $P(\lambda)$ is negligible, then we say $1 - P(\lambda)$ is overwhelming.
- A random variable $V$ has min-entropy $\lambda$, denoted as $H_\infty(V) = \lambda$, if $\max_v \Pr[V = v] = 2^{-\lambda}$, or equivalently $\lambda = -\log \max_v \Pr[V = v]$. If $V$ has min-entropy at least $\lambda$, then $V$ is a $\lambda$ source. Given two random variables $V$ and $W$, the conditional min-entropy of $V$ with respect to $W$ is defined to be $\min_w H_\infty(V|W = w)$, or equivalently $-\log \max_{v,w} \Pr[V = v|W = w]$.

### 2.2 Pairing over Prime-order Groups

A prime-order bilinear group generator is an algorithm $\mathcal{G}_{\mathcal{P}}$ that takes as input a security parameter $\lambda$ and outputs a description $\Gamma = (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g)$ where:

- $\mathbb{G}$ and $\mathbb{G}_T$ are groups of prime-order $p$ with efficiently-computable group laws.
- $g$ is a randomly-chosen generator of $\mathbb{G}$.
- $\hat{e}$ is an efficiently-computable bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, i.e., a map satisfying the following properties:
  - Bilinearity: $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$ for all $a, b \in \mathbb{Z}_p$;
  - Non-degeneracy: $\hat{e}(g, g) \neq 1$.

We say the DLIN (Decision Linear) assumption [7] holds if, for every P.P.T. attacker $\mathcal{A}$, its advantage $|\Pr[b' = b] - \frac{1}{2}|$ is negligible in the game, defined in Fig. 1.

```
1. $\mathit{\Gamma} = (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g) \overset{\$}{\leftarrow} \mathcal{G}_{\mathcal{P}}(\lambda)$
2. $z_1, z_2, z_3, z_4, r \overset{\$}{\leftarrow} \mathbb{Z}_p$
3. $X_0 = (\mathit{\Gamma}, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^{z_3 + z_4})$
   $X_1 = (\mathit{\Gamma}, g^{z_1}, g^{z_2}, g^{z_1 z_3}, g^{z_2 z_4}, g^{r})$
4. $b \in_R \{0,1\}$
5. $b' \overset{\$}{\leftarrow} \mathcal{A}(X_b)$
```

**Fig. 1.** DLIN Assumption

### 2.3 Pairing over Composite-order Groups

A composite-order bilinear group generator is an algorithm $\mathcal{G}_C$ that takes as input a security parameter $\lambda$ and outputs a description $\mathit{\Gamma} = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q)$ where:

- $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order $n = pq$, where $p$ and $q$ are primes, with efficiently computable group laws.
- $g_p$ is a randomly-chosen generator of the subgroup $\mathbb{G}_p$ of order $p$, and $g_q$ is a randomly-chosen generator of the subgroup $\mathbb{G}_q$ of order $q$.
- $\hat{e}$ is an efficiently-computable bilinear pairing $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$, i.e., a map satisfying the following properties for $g \neq 1 \in \mathbb{G}$:
  - Bilinearity: $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$ for all $a, b \in \mathbb{Z}_{pq}$;
  - Non-degeneracy: $\hat{e}(g, g) \neq 1$.

It is worth noting that, instead of setting the order of $\mathbb{G}$ to be $pq$, we can require $\mathcal{G}_C$ to set the order to be the product of multiple prime numbers, e.g. [23–25].

Let $\mathit{\Gamma} = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q)$ be the output by $\mathcal{G}_C(\lambda)$, and $\mathit{\Gamma}^* = (pq, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q)$. We say the Composite-DDH assumption [3, 4] holds if, for every P.P.T. attacker $\mathcal{A}$, its advantage $|\Pr[b' = b] - \frac{1}{2}|$ is negligible in the game, defined in Fig. 2.

```
1. $\mathit{\Gamma} = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q) \overset{\$}{\leftarrow} \mathcal{G}_C(\lambda)$
2. $a_1, a_2, b_1, b_2, b_3, r \overset{\$}{\leftarrow} \mathbb{Z}_{pq}$
3. $\mathit{\Gamma}^* = (pq, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q)$
4. $X_0 = (\mathit{\Gamma}^*, g_p^{a_1} \cdot g_q^{b_1}, g_p^{a_2} \cdot g_q^{b_2}, g_p^{a_1 a_2} \cdot g_q^{b_3})$
   $X_1 = (\mathit{\Gamma}^*, g_p^{a_1} \cdot g_q^{b_1}, g_p^{a_2} \cdot g_q^{b_2}, g_p^{r} \cdot g_q^{b_3})$
5. $b \in_R \{0,1\}$
6. $b' \overset{\$}{\leftarrow} \mathcal{A}(X_b)$
```

**Fig. 2.** Composite-DDH assumption

```
1. $\mathit{\Gamma} = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q) \overset{\$}{\leftarrow} \mathcal{G}_C(\lambda)$
2. $a_1, a_2, a_3, b_1, b_2, b_3, b_4, r \overset{\$}{\leftarrow} \mathbb{Z}_{pq}$
3. $\mathit{\Gamma}^* = (pq, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q)$
4. $X_0 = (\mathit{\Gamma}^*, g_p^{a_1} \cdot g_q^{b_1}, g_p^{a_2} \cdot g_q^{b_2}, g_p^{a_1 a_3} \cdot g_q^{b_3}, g_p^{a_2 a_3} \cdot g_q^{b_4})$
   $X_1 = (\mathit{\Gamma}^*, g_p^{a_1} \cdot g_q^{b_1}, g_p^{a_2} \cdot g_q^{b_2}, g_p^{a_1 a_3} \cdot g_q^{b_3}, g_p^{r} \cdot g_q^{b_4})$
5. $b \in_R \{0,1\}$
6. $b' \overset{\$}{\leftarrow} \mathcal{A}(X_b)$
```

**Fig. 3.** Weak Composite-DDH assumption

We say the Weak Composite-DDH assumption holds if, for every P.P.T. attacker $\mathcal{A}$, its advantage $|\Pr[b' = b] - \frac{1}{2}|$ is negligible in the game, defined in Fig. 3. Both assumptions are strictly weaker than the C3DH assumption by Boneh and Waters [13].

### 2.4 New Assumptions

In [15], Canetti proposed the DDH-II assumption which differs from the standard DDH assumption in that one exponent is chosen from a *wide spread* distribution instead of a uniform one. Damgård, Hazay and, Zottarel [17] showed that this assumption holds in the generic group model, and stated that it is a useful tool in leakage resilient cryptography.

Next, we introduce an analog assumption for bilinear groups. The philosophy is similar to the case of Composite-DDH: although it is trivial to solve DDH-II problem in bilinear groups, adding an additional layer of randomization makes it difficult even with the help of the bilinear map. Formally, we say the Composite-DDH-II assumption holds if, for every P.P.T. attacker $\mathcal{A}$, its advantage $|\Pr[b' = b] - \frac{1}{2}|$ is negligible in the game, defined in Fig. 4. In the game, the distribution $\mathbb{D}$ from the attacker should guarantee that $a_1$ has min-entropy not smaller than $\lambda$, i.e. $a_1$ is *wide spread* according to Canetti [15].

```
1. $\Gamma = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g_p, g_q) \overset{\$}{\leftarrow} \mathcal{G}_C(\lambda)$
2. $\Gamma^* = (pq, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g_p, g_q)$
3. $\mathbb{D} \overset{\$}{\leftarrow} \mathcal{A}(\Gamma^*)$
4. $a_1 \overset{\mathbb{D}}{\leftarrow} \mathbb{Z}_p$
5. $b_1, s_1, s_2, s_3, r \overset{\$}{\leftarrow} \mathbb{Z}_{pq}$
6. $X_0 = (\Gamma^*, g_p^{a_1} \cdot g_q^{s_1}, g_p^{b_1} \cdot g_q^{s_2}, g_p^{a_1 b_1} \cdot g_q^{s_3})$
   $X_1 = (\Gamma^*, g_p^{a_1} \cdot g_q^{s_1}, g_p^{b_1} \cdot g_q^{s_2}, g_p^{r} \cdot g_q^{s_3})$
7. $b \in_R \{0, 1\}$
8. $b' \overset{\$}{\leftarrow} \mathcal{A}(X_b, \mathbb{D})$
```

**Fig. 4.** Composite-DDH-II assumption

```
1. $\Gamma = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g_p, g_q) \overset{\$}{\leftarrow} \mathcal{G}_C(\lambda)$
2. $\Gamma^* = (pq, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g_p, g_q)$
3. $\mathbb{D} \overset{\$}{\leftarrow} \mathcal{A}(\Gamma^*)$
4. $(a_1, \cdots, a_L) \overset{\mathbb{D}}{\leftarrow} \mathbb{Z}_p^L$
5. $b_1, \cdots b_L, r_1, \cdots, r_L, s_1, \cdots, s_L, t_1, \cdots, t_L \overset{\$}{\leftarrow} \mathbb{Z}_{pq}$
6. $X_0 = (\Gamma^*, g_p^{b_1} \cdot g_q^{s_1}, g_p^{a_1 b_1} \cdot g_q^{t_1}, \cdots, g_p^{b_L} \cdot g_q^{s_L}, g_p^{a_L b_L} \cdot g_q^{t_L})$
   $X_1 = (\Gamma^*, g_p^{b_1} \cdot g_q^{s_1}, g_p^{r_1} \cdot g_q^{t_1}, \cdots, g_p^{b_L} \cdot g_q^{t_L}, g_p^{r_L} \cdot g_q^{t_L})$
7. $b \in_R \{0, 1\}$
8. $b' \overset{\$}{\leftarrow} \mathcal{A}(X_b, \mathbb{D})$
```

**Fig. 5.** Correlated Composite-DDH-II assumption

Further, we say the Correlated Composite-DDH-II assumption holds if, for every P.P.T. attacker $\mathcal{A}$, its advantage $|\Pr[b' = b] - \frac{1}{2}|$ is negligible in the game, defined in Fig. 5. In the game, the distribution $\mathbb{D}$ from the attacker should guarantee that $a_i$ for any $1 \leq i \leq L$ has min-entropy not smaller than $\lambda$. Compared to the Composite-DDH-II assumption, on one hand the values $g_p^{a_1}, \cdots, g_p^{a_L}$ are not given to the attacker (not even in the randomized form), but on the other hand the attacker is given multiple incomplete DH pairs. As to these two new assumptions, it is not clear how to reduce one to the other. Nevertheless, we have the following lemma. The proof is in Appendix I.

**Lemma 1.** *Suppose any P.P.T. attacker has at most the advantage $\epsilon$ in the Composite-DDH-II assumption. Then, any P.P.T. attacker has at most the advantage $L \cdot \epsilon$ in the Correlated Composite-DDH-II assumption in the following two scenarios.*

1. *$a_1, a_2, \cdots, a_L$ are independent according to $\mathbb{D}$.*
2. *$a_1 = a_2 = \cdots = a_L$ according to $\mathbb{D}$.*

It is unclear how to reduce these two assumptions to existing standard assumptions. Nevertheless, we can prove their security in the generic group model, as what have been done for the DDH-II assumption in [17]. The proof will appear in the full version of this paper.

## 3 Forward Security Properties for PEKS

A PEKS scheme involves a client, a server, and senders which can be any entity (including the client and the server). Let $\lambda$ be the security parameter, a PEKS scheme consists of the following algorithms.

- KeyGen($\lambda$): Run by the client, this probabilistic algorithm outputs a public/private key pair $(PK, SK)$, where $PK$ should define a message space $\mathcal{W}$.
- Encrypt($x, PK$): Run by a sender, this probabilistic algorithm outputs a ciphertext (or, tag) $C_x$ for a message (or, keyword) $x \in \mathcal{W}$.
- TrapGen($y, SK$): Run by the client, this probabilistic algorithm generates a trapdoor $T_y$ for the message $y \in \mathcal{W}$.
- Test($C_x, T_y, PK$): Run by the server, this deterministic algorithm returns 1 if $x = y$ and 0 otherwise.

**Definition 1.** *A PEKS scheme achieves computational consistency [1], if any P.P.T. attacker $\mathcal{A}$'s advantage* $\Pr[b = 1]$ *is negligible in the attack game, shown in Fig. 6.*

```
1. $(PK, SK) \overset{\$}{\leftarrow}$ KeyGen($\lambda$)
2. $(x, x', state) \overset{\$}{\leftarrow} \mathcal{A}(PK)$
3. Set $b = 1$ iff Test($C_x, T_{x'}, PK$) $= 1$ and $x \neq x'$, where $T_{x'} =$ TrapGen($x', SK$), $C_x =$ Encrypt($x, PK$)
```

**Fig. 6.** Computational Consistency

### 3.1 Security Properties for PEKS

The following ciphertext privacy property is the default property for PEKS from [8].

**Definition 2.** *A PEKS scheme achieves ciphertext privacy if any P.P.T. attacker $\mathcal{A}$'s advantage $|Pr[b' = b] - \frac{1}{2}|$ is negligible in the attack game shown in Fig. 7. In the game, $\mathcal{A}$ is not allowed to query the* TrapGen *oracle with the messages $x_0$ and $x_1$.*

In Definition 3 and Definition 4, we follow the "Real-or-Random" paradigm. This means that $\mathbb{D}_0$ is defined by the attacker at its will, but $\mathbb{D}_1$ always represents an independent uniform distribution for every variable.

The following forward-secure trapdoor unlinkability says that any P.P.T. attacker cannot determine the links among trapdoors as long as the underlying keywords are sampled according to distributions with min-entropy not smaller than $\lambda$. This property is an augmented variant of the strong search pattern privacy property from [3, 4], where the augmentation lies in two aspects.

- One is that the attacker is given $SK$ in Step 4 of the attack game, and this brings in the "forward-secure" flavor.
- The other is that the attacker is allowed to adaptively choose the keyword distributions based on the public parameters, while the challenger samples the keywords uniformly from the keyword space in [3, 4].

**Definition 3.** *A PEKS scheme achieves forward-secure trapdoor unlinkability if any P.P.T. attacker $\mathcal{A}$'s advantage $|Pr[b' = b] - \frac{1}{2}|$ is negligible in the game shown in Fig. 8. In the game, $\mathbb{D}_0$ and $\mathbb{D}_1$ are the joint distributions of $L$ (dependent) $\lambda$-source random variables, which take values in the message space $\mathcal{W}$. The integer $L$ is a polynomial in $\lambda$.*

```
1. $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(\lambda)$
2. $(x_0, x_1, state) \xleftarrow{\$} \mathcal{A}^{\mathsf{TrapGen}}(PK)$
3. $b \in_R \{0,1\},\ C_{x_b} = \mathsf{Encrypt}(x_b, PK)$
4. $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{TrapGen}}(state, C_{x_b})$
```

**Fig. 7.** Ciphertext Privacy

```
1. $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(\lambda)$
2. $(\mathbb{D}_0, \mathbb{D}_1, L, state) \xleftarrow{\$} \mathcal{A}^{\mathsf{TrapGen}}(PK)$
3. $b \in_R \{0,1\},\ \boldsymbol{x}_b \leftarrow \mathbb{D}_b,\ \boldsymbol{T}_b = \mathsf{TrapGen}(\boldsymbol{x}_b, SK)$
4. $b' \xleftarrow{\$} \mathcal{A}(\boxed{\mathsf{SK}}, state, \boldsymbol{T}_b)$
```

**Fig. 8.** Forward-Secure Trapdoor unlinkability

For simplicity, we use $\mathsf{TrapGen}(\boldsymbol{x}_b, SK)$ to denote $(\mathsf{TrapGen}(\boldsymbol{x}_b^{(1)}, SK), \cdots, \mathsf{TrapGen}(\boldsymbol{x}_b^{(L)}, SK))$ in Fig. 8. Such notation is also used in Fig. 9 and properties definitions for IBE.

The following forward-secure function privacy property says that a P.P.T. attacker cannot determine the links among (tag, trapdoor) pairs, as long as the underlying keywords are sampled according to distributions with min-entropy not smaller than $\lambda$ and they are not equal to each other. This property is an augmented variant of the enhanced function privacy property from [11], where the augmentation lies in two aspects. It is worth noting that the property for PEKS is not explicitly defined in [11], but it is implied by their discussions (in fact, they use it to motivate the property for IBE).

- One is that the attacker is given $SK$ in Step 4 of the attack game, and this brings in the "forward-secure" flavor.
- The other is that, in the enhanced function privacy property definition in [11], the conditional min-entropy of $\boldsymbol{x}_0^{(i)}$ given $\boldsymbol{x}_0^{(1)}, \cdots, \boldsymbol{x}_0^{(i-1)}$ is required to be at least $\lambda$, for all $2 \le i \le L$. While, in our definition, we relax this requirement by asking the sampled identities not equal to each other. Based on the fact that the keywords in trapdoors may highly correlated, our requirement is more realistic in practice.

**Definition 4.** *A PEKS scheme achieves forward-secure function privacy if any P.P.T. attacker $\mathcal{A}$'s advantage $|Pr[b' = b] - \frac{1}{2}|$ is negligible in the game shown in Fig. 9. In the game, $\mathbb{D}_0$ and $\mathbb{D}_1$ are defined in the same way as in Definition 3, but with the following restriction: for $(\boldsymbol{x}_0^{(1)}, \boldsymbol{x}_0^{(2)}, \cdots, \boldsymbol{x}_0^{(L)}) \xleftarrow{\mathbb{D}_0} \mathcal{W}^L$ and any $1 \le i \ne j \le L$, the probability $\Pr[\boldsymbol{x}_0^{(i)} = \boldsymbol{x}_0^{(j)}]$ is negligible.*

```
1. $(PK, SK) \xleftarrow{\$} \mathsf{KeyGen}(\lambda)$
2. $(\mathbb{D}_0, \mathbb{D}_1, L, state) \xleftarrow{\$} \mathcal{A}^{\mathsf{TrapGen}}(PK)$
3. $b \in_R \{0,1\},\ \boldsymbol{x}_b \leftarrow \mathbb{D}_b,\ \boldsymbol{T}_b = \mathsf{TrapGen}(\boldsymbol{x}_b, SK),\ \boldsymbol{C}_b = \mathsf{Encrypt}(\boldsymbol{x}_b, PK)$
4. $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{TrapGen},\mathsf{Encrypt}}(\boxed{\mathsf{SK}}, state, \boldsymbol{T}_b, \boldsymbol{C}_b)$
```

**Fig. 9.** Forward-Secure Function Privacy

In Step 4 of the above game, in a TrapGen oracle query, the attacker has an index $1 \leq i \leq L$ as input and receives $\mathsf{TrapGen}(x_b^{(i)}, SK)$. In an Encrypt oracle query, the attacker has an index $1 \leq j \leq L$ as input and receives $\mathsf{Encrypt}(x_b^{(j)}, PK)$. Note that the sampled identities $x_0^{(1)}, \cdots, x_0^{(L)}$ are not allowed to be equal in the attack game. Hence, both oracles are necessary even if the attacker possesses $SK$, because the attacker may not be able to generate a new trapdoor or tag by re-randomizing an existing one. These two oracles faithfully reflect the fact that, in practice, the attacker (i.e. the server) may receive many tags and trapdoors for the same keyword.

*Remark 1.* Similar to the argument in [3], the forward-secure trapdoor unlinkability property and the forward-secure function privacy property do not imply each other. We skip the details in this paper.

### 3.2 Analysis of Nishioka Scheme

In [25], Nishioka modeled trapdoor unlinkability for a very restricted setting: the attacker is non-adaptive, the unlinkability is only for two trapdoors, and the model seems to be selective since the challenge keywords are chosen before the generation of other parameters (in the SPP experiment). Relying on bilinear groups of composite-order $pqw$, Nishioka constructed the following scheme (referred to as Instance 3 in [25]) and proved the ciphertext privacy and trapdoor unlinkability properties under some "non-standard" assumptions (as stated by Nishioka). In the following, we show that the scheme actually achieves more than what Nishioka has expected.

| $\mathsf{KeyGen}(\lambda)$ | $\mathsf{TrapGen}(y, SK)$ |
|---|---|
| $(p, q, w, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q, g_w) \xleftarrow{\$} \mathcal{G}_C{}^*(\lambda)$<br>$g_q^\dagger \xleftarrow{\$} \mathbb{G}_q,\ g = g_p \cdot g_q^\dagger$<br>$\mathcal{W} = \{0,1\}^*,\ \mathsf{H} : \{0,1\}^* \to \mathbb{G}_p$<br>$PK = (pqw, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_q, g_w, g, \mathcal{W}, \mathsf{H})$<br>$SK = (PK, g_p)$ | $r_1 \xleftarrow{\$} \mathbb{Z}_{pqw}$<br>$g'_w, g''_w \xleftarrow{\$} \mathbb{G}_w$<br>$T_1 = g_p^{r_1} \cdot g'_w$<br>$T_2 = \mathsf{H}(y)^{r_1} \cdot g''_w$<br>$T_y = (T_1, T_2)$ |
| $\mathsf{Encrypt}(x, PK)$ | $\mathsf{Test}(C_x, T_y, PK)$ |
| $r_2 \xleftarrow{\$} \mathbb{Z}_{pqw},\ g'_q, g''_q \xleftarrow{\$} \mathbb{G}_q$<br>$C_1 = g^{r_2} \cdot g'_q, C_2 = \mathsf{H}(x)^{r_2} \cdot g''_q$<br>$C_x = (C_1, C_2)$ | if $\hat{e}(T_1, C_2) = \hat{e}(T_2, C_1)$, output 1<br>otherwise, output 0 |

Note that we use the notation $\mathcal{G}_C{}^*$ to emphasize the fact that this algorithm generates bilinear groups of order $pqw$, in contrast to the definition of $\mathcal{G}_C$ in Section 2.3. A minor remark is that $r_1$ is defined to be $r_1 \xleftarrow{\$} \mathbb{Z}_p$ for the TrapGen algorithm in [25], but it is clear that $r_1 \xleftarrow{\$} \mathbb{Z}_{pqw}$ makes the scheme work in the same way.

In Definition 3 and 4, we assume the attacker to be fully adaptive in the sense that it can choose the distribution $\mathbb{D}_0$ based on everything. An immediate relaxation on these definitions is to make the attacker *RO-non-adaptive*, which means that the attacker can choose the distribution $\mathbb{D}_0$ based on everything except for the random oracle (i.e. the hash function). In practice, the keywords in search queries might be related to the system parameters in some manner, but it is hard to imagine a scenario where the keywords would depend on the behavior of a random function. We argue that the relaxation is minimal and reasonable.

Next, we prove that the scheme achieves forward-secure properties for RO-non-adaptive attackers in the random oracle model. The proofs appear in Appendix II. It is worth mentioning that Theorem 1 is implied by the discussion in [11].

**Theorem 1.** *The scheme achieves RO-non-adaptive forward-secure function privacy property in the random oracle model.*

**Theorem 2.** *The scheme achieves RO-non-adaptive forward-secure trapdoor unlinkability property based on the Weak Composite-DDH assumption in the random oracle model.*

A natural following-up question is whether or not the scheme is (adaptively) forward-secure in the random oracle model. Referring to the forward-secure trapdoor unlinkability attack game, as shown in Fig. 8, the attacker is required to distinguish the same pairs as in the the proof of Theorem 2, with the exception that the distribution $\mathbb{D}_0$ is defined by the attacker based on the knowledge of $\mathsf{H}$. It is straightforward to verify that the attacker's advantage is negligible based on the correlated Composite-DDH-II assumption, defined in Section 2.4. However, it is very tricky for the forward-secure function privacy property. Referring to the attack game, as shown in Fig. 9, the attacker's mission is to distinguish the following two pairs.

$$\boxed{\text{b=0}}: \quad ((g_p^{r_1^{(1)}} \cdot g_w'^{(1)},\ \mathsf{H}(\boldsymbol{x}_0^{(1)})^{r_1^{(1)}} \cdot g_w''^{(1)},\ g_p^{r_2^{(1)}} \cdot g_q'^{(1)}, \mathsf{H}(\boldsymbol{x}_0^{(1)})^{r_2^{(1)}} \cdot g_q''^{(1)}),\ \cdots,$$
$$(g_p^{r_1^{(L)}} \cdot g_w'^{(L)},\ \mathsf{H}(\boldsymbol{x}_0^{(L)})^{r_1^{(L)}} \cdot g_w''^{(L)}),\ g_p^{r_2^{(L)}} \cdot g_q'^{(L)},\ \mathsf{H}(\boldsymbol{x}_0^{(L)})^{r_2^{(L)}} \cdot g_q''^{(L)}))$$

$$\boxed{\text{b=1}}: \quad ((g_p^{r_1^{(1)}} \cdot g_w'^{(1)},\ \mathsf{H}(\boldsymbol{x}_1^{(1)})^{r_1^{(1)}} \cdot g_w''^{(1)},\ g_p^{r_2^{(1)}} \cdot g_q'^{(1)},\ \mathsf{H}(\boldsymbol{x}_1^{(1)})^{r_2^{(1)}} \cdot g_q''^{(1)}),\ \cdots,$$
$$(g_p^{r_1^{(L)}} \cdot g_w'^{(L)},\ \mathsf{H}(\boldsymbol{x}_1^{(L)})^{r_1^{(L)}} \cdot g_w''^{(L)},\ g_p^{r_2^{(L)}} \cdot g_q'^{(L)},\ \mathsf{H}(\boldsymbol{x}_1^{(L)})^{r_2^{(L)}} \cdot g_q''^{(L)}))$$

We are not able to reduce this assumption to the correlated Composite-DDH-II assumption or any other assumption, even though we might be able to prove that distinguishing the two pairs is hard in the generic group model. We leave a further investigation of this as a future work.

## 4 IBE and its Security Properties

An IBE scheme is specified by four algorithms $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$, defined as follows.

- $\mathsf{Setup}(\lambda)$: On input the security parameter $\lambda$, this probabilistic algorithm returns a master secret key *Msk* and public parameters *params*, which should define a message space $\mathcal{M}$ and identity space $\mathcal{I}$.
- $\mathsf{Extract}(Msk, id)$: On input a master secret key *Msk* and a public key $id \in \mathcal{I}$, this probabilistic algorithm outputs a secret key $sk_{id}$.
- $\mathsf{Enc}(m, id)$: On input a message $m \in \mathcal{M}$ and a public key $id \in \mathcal{I}$, this probabilistic algorithm outputs a ciphertext $C$.
- $\mathsf{Dec}(C, sk_{id})$: On input a ciphertext $C$ and a secret key $sk_{id}$, this deterministic algorithm outputs a message $m$ or an error symbol $\perp$.

### 4.1 Generic Transformation from IBE to PEKS

Given an IBE scheme $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$, the generic transformation works as follows [1]. Note that the message space $\mathcal{W}$ of the resulted PEKS scheme is the public-key space $\mathcal{I}$ of the original IBE scheme.

1. $(Msk, params) = \mathsf{Setup}(\lambda)$
2. $sk_{id} = \mathsf{Extract}(Msk, id)$
3. $C = \mathsf{Enc}(m, id)$
4. $\mathsf{Dec}(C,\ sk_{id}) = m$ or $\perp$

**Fig. 10.** Original IBE

1. $\boxed{\mathsf{KeyGen}(\lambda)} = \mathsf{Setup}(\lambda)$
2. $\boxed{\mathsf{Encrypt}(x, PK)} = (m,\ \mathsf{Enc}(m, x))$, where $m \in_R \mathcal{M}$
3. $\boxed{\mathsf{TrapGen}(y, SK)} = \mathsf{Extract}(Msk, y)$
4. $\boxed{\mathsf{Test}(C_x, T_y, PK)} = 1$ iff $m = \mathsf{Dec}(\mathsf{Enc}(m, x), T_y)$

**Fig. 11.** Resulted PEKS

### 4.2 Security Properties of IBE

In this subsection, we present four IBE security properties, which respectively lead to the four desirable properties for the resulted PEKS through the generic transformation. Their correspondence is summarized in the following table.

| PEKS Properties | IBE Properties |
|---|---|
| computational consistency | IND-CPA |
| ciphertext privacy | anonymity |
| forward-secure trapdoor unlinkability | forward-secure key unlinkability |
| forward-secure function privacy | forward-secure function privacy |

In Definition 6 and Definition 7, we follow the "Real-or-Random" paradigm in all relevant attack games. This means that $\mathbb{D}_0$ is defined by the attacker at its will, but $\mathbb{D}_1$ always represents an independent uniform distribution for every variable.

**Definition 5.** *An IBE scheme achieves IND-CPA security if any P.P.T. attacker $\mathcal{A}$'s advantage $|Pr[b' = b] - \frac{1}{2}|$ is negligible in the attack game shown in Fig. 12. In the game, the attacker is not allowed to query the* Extract *oracle with $id^*$. An IBE scheme achieves anonymity if any P.P.T. attacker $\mathcal{A}$'s advantage $|Pr[b' = b] - \frac{1}{2}|$ is negligible in the game shown in Fig. 13. In the game, the attacker is not allowed to query the* Extract *oracle with $id_0$ and $id_1$.*

```
1. $(Msk, params) \xleftarrow{\$} \mathsf{Setup}(\lambda)$
2. $(m_0, m_1, id^*, state) \xleftarrow{\$} \mathcal{A}^{\mathsf{Extract}}(params)$
3. $b \in_R \{0,1\}$,  $C_b = \mathsf{Enc}(m_b, id^*)$
4. $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{Extract}}(state, C_b)$
```

**Fig. 12.** IND-CPA

```
1. $(Msk, params) \xleftarrow{\$} \mathsf{Setup}(\lambda)$
2. $(m, id_0, id_1, state) \xleftarrow{\$} \mathcal{A}^{\mathsf{Extract}}(params)$
3. $b \in_R \{0,1\}$,  $C_b = \mathsf{Enc}(m, id_b)$
4. $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{Extract}}(state, C_b)$
```

**Fig. 13.** Anonymity

The following forward-secure key unlinkability property says that any P.P.T. attacker cannot determine the links among private keys as long as the underlying identities are sampled according to distributions with min-entropy not smaller than $\lambda$. This property is an augmented variant of the strong key unlinkability property from [3, 4], where the augmentation lies in two aspects.

– One is that the attacker is given *Msk* in Step 4 of the attack game, and this brings in the "forward-secure" flavor.
– The other is that the attacker is allowed to adaptively choose the identity distribution $\mathbb{D}_0$ based on the public parameters, while the challenger samples the identities uniformly at random (according to certain patterns defined by the attacker) from the identity space in [3, 4].

**Definition 6.** *An IBE scheme achieves forward-secure key unlinkability if any P.P.T. attacker $\mathcal{A}$'s advantage $|Pr[b' = b] - \frac{1}{2}|$ is negligible in the game shown in Fig. 14. In the game, $\mathbb{D}_0$ and $\mathbb{D}_1$ are the joint distribution of L (dependent) $\lambda$-source random variables, which take values in the identity space $\mathcal{I}$. The integer L is a polynomial in $\lambda$.*

```
1. $(Msk, params) \xleftarrow{\$} \mathsf{Setup}(\lambda)$
2. $(\mathbb{D}_0, \mathbb{D}_1, L, state) \xleftarrow{\$} \mathcal{A}^{\mathsf{Extract}}(params)$
3. $b \in_R \{0,1\}$,  $\boldsymbol{id}_b \leftarrow \mathbb{D}_b$,  $\boldsymbol{sk}_b = \mathsf{Extract}(\boldsymbol{id}_b, Msk)$
4. $b' \xleftarrow{\$} \mathcal{A}(\boxed{\text{Msk}}, state, \boldsymbol{sk}_b)$
```

**Fig. 14.** Forward-Secure Key Unlinkability

The following forward-secure function privacy property says that any P.P.T. attacker cannot determine the links among (private key, ciphertext) pairs, as long as the underlying identities are sampled according to distributions with min-entropy not smaller than $\lambda$ and they are not equal to each other. This property is an augmented variant of the enhanced function privacy property from [11], where the augmentation lies in two aspects.

- One is that the attacker is given $Msk$ in Step 4 of the attack game, and this brings in the "forward-secure" flavor.
- The other is that, in the enhanced function privacy property definition from [11], the conditional min-entropy of $\boldsymbol{id}_0^{(i)}$ given $\boldsymbol{id}_0^{(1)}, \cdots, \boldsymbol{id}_0^{(i-1)}$ is required to be at least $\lambda$, for all $2 \leq i \leq L$. In our definition, we relax this requirement by only asking the sampled identities not equal to each other.

**Definition 7.** *An IBE scheme achieves* <u>*forward-secure function privacy*</u> *if any P.P.T. attacker* $\mathcal{A}$*'s advantage* $|Pr[b' = b] - \frac{1}{2}|$ *is negligible in the game shown in Fig. 15. In the game,* $\mathbb{D}_0$ *and* $\mathbb{D}_1$ *are defined in the same way as in Definition 6, but with the following restriction: for* $(\boldsymbol{id}_0^{(1)}, \boldsymbol{id}_0^{(2)}, \cdots, \boldsymbol{id}_0^{(L)}) \xleftarrow{\mathbb{D}_0} \mathcal{I}^L$ *and any* $1 \leq i \neq j \leq L$*, the probability* $\Pr[\boldsymbol{id}_0^{(i)} = \boldsymbol{id}_0^{(j)}]$ *is negligible.*

1. $(PK, SK) \xleftarrow{\$} \mathsf{Setup}(\lambda)$
2. $(\mathbb{D}_0, \mathbb{D}_1, L, state) \xleftarrow{\$} \mathcal{A}^{\mathsf{Extract}}(params)$
3. $b \in_R \{0, 1\}$, $\boldsymbol{id}_b \leftarrow \mathbb{D}_b$, $\boldsymbol{sk}_b = \mathsf{Extract}(\boldsymbol{id}_b, Msk)$, $\boldsymbol{m}_b \xleftarrow{\$} \mathcal{M}^L$, $\boldsymbol{C}_b = \mathsf{Enc}(\boldsymbol{m}_b, \boldsymbol{id}_b)$
4. $b' \xleftarrow{\$} \mathcal{A}^{\mathsf{Extract},\mathsf{Enc}}(\boxed{\text{Msk}}, state, \boldsymbol{sk}_b, \boldsymbol{C}_b)$

**Fig. 15.** Forward-Secure Function Privacy

For simplicity, we use $\mathsf{Enc}(\boldsymbol{m}_b, \boldsymbol{id}_b)$ to denote $(\mathsf{Enc}(\boldsymbol{m}_b^{(1)}, \boldsymbol{id}_b^{(1)}), \cdots, \mathsf{Enc}(\boldsymbol{m}_b^{(L)}, \boldsymbol{id}_b)^{(L)})$ in Fig. 15.

In Step 4 of the above game, in an Extract oracle query, the attacker has an index $1 \leq i \leq L$ as input and receives $\mathsf{Extract}(Msk, \boldsymbol{id}_b^{(i)})$. In an Enc oracle the attacker has an index $1 \leq j \leq L$ as input and receives $\mathsf{Enc}(m, \boldsymbol{id}_b^{(j)})$. It is obvious that both oracles are necessary even if the attacker possesses $Msk$, because the attacker may not be able to generate a new secret key or a ciphertext by re-randomizing an existing one.

## 5 Existing Function-Private IBE Schemes

In this section, we first show that the fully-secure IBE scheme by Boneh, Raghunathan, and Segev [10, 11] does not achieve the forward-secure function privacy. We then prove that a scheme by Arriaga, Tang, and Ryan [3, 4] achieves RO-non-adaptive forward-secure function privacy and forward-secure key unlinkability in the random oracle model.

### 5.1 Boneh-Raghunathan-Segev $\mathcal{IBE}_{\mathsf{DLIN2}}$ Scheme

The following scheme is referred to as $\mathcal{IBE}_{\mathsf{DLIN2}}$ in [11]. It has been proven fully-secure with respect to enhanced function privacy (under the definition in [11]) based on the DLIN assumption in the standard model.

| <u>$\mathsf{Setup}(\lambda)$</u> | <u>$\mathsf{Extract}(Msk, \boldsymbol{id})$</u> |
|---|---|
| $\Gamma = (\mathbb{G}, \mathbb{G}_T, \hat{e}, g, p) = \mathcal{G}_{\mathcal{P}}(\lambda)$<br>$\boldsymbol{A}_0, \boldsymbol{B}, \boldsymbol{A}_1, \cdots, \boldsymbol{A}_n \xleftarrow{\$} \mathbb{Z}_p^{2\times m}$<br>$\boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_p^2$, $\mathcal{M} = \mathbb{G}_T$, $\mathcal{I} = \{0, 1\}^n$<br>$Msk = (\boldsymbol{A}_0, \boldsymbol{B}, \boldsymbol{A}_1, \cdots, \boldsymbol{A}_n, \boldsymbol{u})$<br>$params = (\Gamma, g^{\boldsymbol{A}_0}, \boldsymbol{B}, g^{\boldsymbol{A}_1}, \cdots, g^{\boldsymbol{A}_n}, g^{\boldsymbol{u}}, \mathcal{M}, \mathcal{I})$ | $\boldsymbol{id} = (id_1, id_2, \cdots, id_n) \in \{0, 1\}^n$<br>$\boldsymbol{S} \xleftarrow{\$} \mathbb{Z}_p^{m\times 2}$<br>$\boldsymbol{F_{id,S}} = [\boldsymbol{A}_0 \vert \boldsymbol{BS} + (\sum_{1\leq j\leq n} id_j \boldsymbol{A}_j)\boldsymbol{S}]$<br>$\boldsymbol{v} \xleftarrow{\$} \{\boldsymbol{x} \mid \boldsymbol{F_{id,S}} \cdot \boldsymbol{x} = \boldsymbol{u} \pmod p\}$<br>$\boldsymbol{z} = g^{\boldsymbol{v}} \in \mathbb{G}^{m+2}$, $sk_{\boldsymbol{id}} = (\boldsymbol{S}, \boldsymbol{z})$ |
| <u>$\mathsf{Enc}(m, \boldsymbol{id})$</u> | <u>$\mathsf{Dec}(C, sk_{\boldsymbol{id}})$</u> |
| $\boldsymbol{id} = (id_1, id_2, \cdots, id_n) \in \{0, 1\}^n$, $m \in \mathbb{G}_T$<br>$\boldsymbol{D}(\boldsymbol{id}) = \sum_{1\leq j\leq n} id_j \boldsymbol{A_j}$, $\boldsymbol{r} \xleftarrow{\$} \mathbb{Z}_p^2$<br>$\boldsymbol{c_0}^T = g^{\boldsymbol{r}^T \boldsymbol{A_0}}$, $\boldsymbol{c_1}^T = g^{\boldsymbol{r}^T[\boldsymbol{B}+\boldsymbol{D}(\boldsymbol{id})]}$, $c_2 = m \cdot \hat{e}(g, g)^{\boldsymbol{r}^T \boldsymbol{u}}$<br>$C = (\boldsymbol{c_0}, \boldsymbol{c_1}, c_2)$ | $\boldsymbol{d}^T = [\boldsymbol{c_0}^T \vert (\boldsymbol{c_1}^T)^{\boldsymbol{S}}] = g^{\boldsymbol{r}^T \boldsymbol{F_{id,S}}}$<br>$\hat{e}(\boldsymbol{d}, \boldsymbol{z}) = \hat{e}(g, g)^{\boldsymbol{r}^T(\boldsymbol{F_{id,S}} \cdot \boldsymbol{v})} = \hat{e}(g, g)^{\boldsymbol{r}^T \boldsymbol{u}}$<br>$m = c_2 \cdot \hat{e}(\boldsymbol{d}, \boldsymbol{z})^{-1}$ |

We show that this scheme does not achieve forward-secure function privacy, namely an attacker wins the attack game in Fig. 15 with overwhelming probability. The following attack makes use of the fact that, with $Msk$, the attacker can transform a ciphertext under an identity $\boldsymbol{id}$ into a ciphertext under another identity $\boldsymbol{id}'$, for some carefully chosen $\boldsymbol{id}$ and $\boldsymbol{id}'$.

To mount an attack, in step 2 and 4 of the game, the attacker performs as follows.

- In step 2, the attacker sets $L = 2$, which means $\mathbb{D}_0$ and $\mathbb{D}_1$ are the joint distribution of two identities $(\boldsymbol{id}^{(1)}, \boldsymbol{id}^{(2)})$. For $\mathbb{D}_0$, the attacker sets $\boldsymbol{id}^{(1)} = (id_1, id_2, \cdots, id_n)$ as follows: $(id_1, id_2, \cdots, id_{n-1}) \xleftarrow{\$} \{0,1\}^{n-1}$ and $id_n = 0$. The attacker sets $\boldsymbol{id}^{(2)} = \boldsymbol{id}^{(1)}$ except that $id_n = 1$.
- In step 4, the attacker firstly obtains $Msk = (A_0, \boldsymbol{B}, \boldsymbol{A}_1, \cdots, \boldsymbol{A}_n, \boldsymbol{u})$. Then, the attacker computes $\boldsymbol{X} \in \mathbb{Z}_p^{m\times m}$ such that $\boldsymbol{A_n} = \boldsymbol{A_0 X}$. Recall that the challenge is $(\boldsymbol{sk}_b, \boldsymbol{C}_b)$. Suppose that the first ciphertext in $\boldsymbol{C}_b$ is $C = (\boldsymbol{c}_0, \boldsymbol{c}_1, c_2)$, which is in the following form:

$$\boldsymbol{c_0}^T = g^{\boldsymbol{r}^T \boldsymbol{A}_0}, \boldsymbol{c_1}^T = g^{\boldsymbol{r}^T[\boldsymbol{B}+\boldsymbol{D}(\boldsymbol{id}^{(1)})]}, c_2 = m \cdot \hat{e}(g,g)^{\boldsymbol{r}^T \boldsymbol{u}}.$$

The attacker now generates a new ciphertext $C' = (\boldsymbol{c_0}, \boldsymbol{c}'_1, c_2)$, where

$$\begin{aligned} \boldsymbol{c}'^{T}_{\boldsymbol{1}} &= \boldsymbol{c_1}^T \otimes (\boldsymbol{c_0}^T)^{\boldsymbol{X}} \\ &= g^{\boldsymbol{r}^T[\boldsymbol{B}+\boldsymbol{D}(\boldsymbol{id}^{(1)})]} \otimes g^{\boldsymbol{r}^T \boldsymbol{A}_0 \boldsymbol{X}} \\ &= g^{\boldsymbol{r}^T[\boldsymbol{B}+\boldsymbol{D}(\boldsymbol{id}^{(1)})]} \otimes g^{\boldsymbol{r}^T \boldsymbol{A}_n} \end{aligned}$$

Let the secret keys in the challenge $\boldsymbol{sk}_b$ be denoted as $(sk_{\boldsymbol{id}^{(1)}}, sk_{\boldsymbol{id}^{(2)}})$. The attacker outputs 0 if $\mathsf{Dec}(C', sk_{\boldsymbol{id}^{(2)}}) = \mathsf{Dec}(C, sk_{\boldsymbol{id}^{(1)}})$, and outputs 1 otherwise.

Recall that, $\otimes$ is an operator for pairwise group operations between two the new vectors or matrices. It is clear that $\boldsymbol{c}'^{T}_{\boldsymbol{1}} = g^{\boldsymbol{r}^T[\boldsymbol{B}+\boldsymbol{D}(\boldsymbol{id}^{(2)})]}$ if $(\boldsymbol{id}^{(1)}, \boldsymbol{id}^{(2)})$ is sampled according to $\mathbb{D}_0$, then $C'$ is a ciphertext for $\mathsf{Dec}(C, sk_{\boldsymbol{id}^{(1)}})$ under the identity $\boldsymbol{id}^{(2)}$. But, this equality holds with a negligible probability if the identities are sampled according to $\mathbb{D}_1$. The attack works.

### 5.2 Arriaga-Tang-Ryan IBE Scheme

The following scheme was proposed by Arriaga, Tang, and Ryan in [3, 4], based on an anonymous IBE scheme by Boyen and Waters [14]. This scheme has been proven secure with respect to the strong key unlinkability property (under the definition in [3, 4]) in the random oracle model. Compared with our forward-secure key unlinkability property, the definition from [3, 4] is weaker in three aspects: (1) the attacker is not allowed to adaptively choose the identity distribution $\mathbb{D}_0$ and it can only specify the identity patterns (i.e. which identities are equal); (2) according to the patterns, the challenger samples the identities uniformly at random from the identity space; (3) there is no forward security.

| $\underline{\mathsf{Setup}(\lambda)}$ | $\underline{\mathsf{Extract}(Msk, id)}$ |
|---|---|
| $\Gamma = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g_p, g_q) = \mathcal{G}_C(\lambda)$ | $r \xleftarrow{\$} \mathbb{Z}_n$ |
| $\Gamma^* = (n = pq, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g_p, g_q)$ | $x_0, x_1, x_2 \xleftarrow{\$} \mathbb{G}_q$ |
| $x, t_1, t_2 \xleftarrow{\$} \mathbb{Z}_n$ | $d_0 = x_0 \cdot g_p^{rt_1t_2}$ |
| $\Omega = \hat{e}(g_p, g_p)^{xt_1t_2}, v_1 = g_p^{t_1}, v_2 = g_p^{t_2}$ | $d_1 = x_1 \cdot g_p^{-xt_2} \cdot \mathsf{H}(id)^{-rt_2}$ |
| $\mathcal{M} = \mathbb{G}_T, \mathcal{I} = \{0,1\}^*, \mathsf{H} : \{0,1\}^* \to \mathbb{G}_p$ | $d_2 = x_2 \cdot g_p^{-xt_1} \cdot \mathsf{H}(id)^{-rt_1}$ |
| $Msk = (x, t_1, t_2), params = (\Gamma^*, \Omega, v_1, v_2, \mathcal{M}, \mathcal{I}, \mathsf{H})$ | $sk_{id} = (d_0, d_1, d_2)$ |
| $\underline{\mathsf{Enc}(m, id)}$ | $\underline{\mathsf{Dec}(C, sk_{id})}$ |
| $s, s_1 \in_R \mathbb{Z}_n$ | $e_0 = \hat{e}(c_0, d_0), e_1 = \hat{e}(c_1, d_1)$ |
| $\hat{c} = \Omega^s m$, $c_0 = \mathsf{H}(id)^s$, $c_1 = v_1^{s-s_1}$, and $c_2 = v_2^{s_1}$ | $e_2 = \hat{e}(c_2, d_2)$, |
| $C = (\hat{c}, c_0, c_1, c_2)$ | $m = \hat{c} \cdot e_0 \cdot e_1 \cdot e_2$ |

Similar to the discussions in Section 3.2, an immediate relaxation on Definition 6 and 7 is to make the attacker *RO-non-adaptive*, which means that the attacker can choose the distribution $\mathbb{D}_0$ based on everything except for the random oracle (i.e. the hash function). Next, we prove that the above scheme is secure under Definition 6 and 7 for a RO-non-adaptive attacker. This result is much stronger that what has been proven in [3, 4].

**Theorem 3.** *The scheme achieves RO-non-adaptive forward-secure function privacy property in the random oracle model.*

The proof of Theorem 3 is exactly the same as that for Theorem 1, and it is implied by Theorem 6.1 from [11]. Next, we prove the RO-non-adaptive forward-secure key unlinkability property. This result is an improvement to Lemma 3 from [3], and it relies on Weak Composite-DDH assumption instead of Composite-DDH assumption. The proof appears in Appendix III.

**Theorem 4.** *The scheme achieves RO-non-adaptive forward-secure key unlinkability property based on the Weak Composite-DDH assumption in the random oracle model.*

With these results, a following-up question is whether or not the scheme is (adaptively) forward-secure. Referring to the forward-secure key unlinkability attack game, as shown in Fig. 14, it is straightforward to verify that the attacker's advantage is negligible based on the correlated Composite-DDH-II assumption, similar to the proof of Theorem 6 in next section. However, it is very tricky for the forward-secure function privacy property. Referring to the attack game, as shown in Fig. 15, the attacker's mission is to distinguish the following two pairs.

$$\boxed{\text{b=0}}: \ ((g_p^{r_1^{(1)}}, \ \mathsf{H}(\boldsymbol{id}_0^{(1)})^{r_1^{(1)}}), \ (g_p^{r_1^{(2)}}, \ \mathsf{H}(\boldsymbol{id}_0^{(2)})^{r_1^{(2)}}), \ \cdots, (g_p^{r_1^{(L)}}, \ \mathsf{H}(\boldsymbol{id}_0^{(L)})^{r_1^{(L)}}))$$

$$\boxed{\text{b=1}}: \ ((g_p^{r_1^{(1)}}, \ \mathsf{H}(\boldsymbol{id}_1^{(1)})^{r_1^{(1)}}), \ (g_p^{r_1^{(2)}}, \ \mathsf{H}(\boldsymbol{id}_1^{(2)})^{r_1^{(2)}}), \ \cdots, (g_p^{r_1^{(L)}}, \ \mathsf{H}(\boldsymbol{id}_1^{(L)})^{r_1^{(L)}}))$$

This seems to be strictly easier for the attacker than in the case of correlated Composite-DDH-II assumption, because these pairs are not randomized by anything from $\mathbb{G}_q$.

## 6 Forward-Secure IBE Construction

In [10, 11], Boneh, Raghunathan and Segev indicated that function privacy against non-adaptive attackers is almost trivial to achieve in the random oracle model. In Section 5.2, we showed that forward-secure function privacy can be achieved against RO-non-adaptive attackers in the random oracle model. But, it becomes very tricky in the presence of adaptive attackers. If an attacker defines the distributions based on the behavior of random oracle (i.e. hash function), then we cannot easily reduce the security to standard hardness assumptions, as shown in the end of last section. In Section **??** and 5.1, our attacks show that the elegant "extract-combine-augment" approach from [10, 11] falls short of achieving our forward-secure function privacy property, intuitively because it introduces "good" algebraic structures into the ciphertexts and allows the attacker to mount an attack accordingly.

In its implementation, the "extract-combine-augment" approach requires specific modifications to both encryption and decryption algorithms in the underlying IBE scheme. This may not be an easy task to achieve, in particularly to achieve our forward security properties. In the following, we introduce the concept of deterministic scrambler, which serves as a building block to pre-process identities for any IBE scheme. Similar to the "extract" step in the "extract-combine-augment" approach, a deterministic scrambler scheme eliminates the dependency relationships among correlated random variables while it does not require any further change to the underlying IBE scheme. As an example application, we extend the Boyen-Waters anonymous IBE scheme and prove its security properties.

### 6.1 Deterministic Scrambler

A deterministic scrambler scheme is a cryptographic primitive to map inputs from a set $\mathcal{X}$ to outputs from another set $\mathcal{Y}$. Formally, it consists of two algorithms.

- $\mathsf{DS.init}(\lambda)$: this probabilistic algorithm returns the public parameter *param*.
- $\mathsf{DS.enc}(x, param)$: this deterministic algorithm returns a scrambled output $y \in \mathcal{Y}$.

A deterministic scrambler scheme is sound if it is injective with overwhelming probability. Formally, the probability that, an attacker can find $x_1 \neq x_2 \in \mathcal{X}$ such that $\mathsf{DS.enc}(x_1, param) = \mathsf{DS.enc}(x_2, param)$, is negligible.

**Definition 8.** *A deterministic scrambler scheme is secure if the attacker's advantage* $|\Pr[b' = b] - \frac{1}{2}|$ *is negligible in the attack game, shown in Fig. 16. In the game,* $\mathbb{D}_0$ *and* $\mathbb{D}_1$ *are the joint distributions of* $L$ *(dependent)* $\lambda$*-source random variables, which take values in the space* $\mathcal{X}$*. It is required that, for* $(\boldsymbol{x}_0^{(1)}, \boldsymbol{x}_0^{(2)}, \cdots, \boldsymbol{x}_0^{(L)}) \overset{\mathbb{D}_0}{\leftarrow} \mathcal{X}^L$ *and any* $1 \le i \ne j \le L$*, the probability* $\Pr[\boldsymbol{x}_0^{(i)} = \boldsymbol{x}_0^{(j)}]$ *is negligible. The integer* $L$ *is a polynomial in* $\lambda$*.*

```
1. $param \leftarrow \mathsf{DS.init}(\lambda)$
2. $(\mathbb{D}_0, \mathbb{D}_1, L, state) \overset{\$}{\leftarrow} \mathcal{A}(param)$
3. $b \in_R \{0,1\},\ \boldsymbol{x}_b \overset{\mathbb{D}_b}{\leftarrow} \mathcal{X}^L,\ \boldsymbol{y}_b = \mathsf{DS.enc}(\boldsymbol{x}_b, param)$
4. $b' \overset{\$}{\leftarrow} \mathcal{A}(state, \boldsymbol{y}_b)$
```

**Fig. 16.** Deterministic Scrambler Security

Note that, we follow the "Real-or-Random" paradigm, which means that $\mathbb{D}_0$ is defined by the attacker at its will, but $\mathbb{D}_1$ represents an independent uniform distribution for every variable.

The concept of deterministic scrambler is closely related to that of correlated-input secure hash functions by Goyal, O'Neill and Rao [19]. Its soundness property is indeed the standard collision resistance property of hash function, and its security property is related to the correlated-input pseudorandomness property. Unfortunately, only selective security properties have been considered in [19] and the constructions assume certain specific correlations among the inputs. The concept of deterministic scrambler is also related to that of unseeded deterministic extractors [28]. But, they have subtle differences: deterministic scrambler generates indistinguishable outputs for correlated inputs and random inputs, while unseeded deterministic extractor generates outputs which are indistinguishable from random strings. It may seem straightforward to instantiate a deterministic scrambler scheme from an unseeded deterministic extractor scheme. However, the existing constructions for unseeded deterministic extractors usually put strict constraints on the sources (or, inputs), hence it is unclear how to instantiate a deterministic scrambler scheme from them.

Yet another related concept is deterministic encryption [5] since both primitives aim at hiding the dependent relationships of their inputs. With respect to their functionalities, deterministic scrambler is simpler because it does not need a *decryption* function. With respect to security, deterministic scrambler is stronger in the sense that it assumes adaptive attackers while the security definitions in [5] and most following-ups assume non-adaptive attackers. Very recently, Raghunathan, Segev and Vadhan have considered adaptive security for deterministic encryption schemes [26]. Based on their scheme, we can instantiate a deterministic scrambler scheme in a straightforward manner: set *param* to be the public key and set the function $\mathsf{DS.enc}$ be the encryption function. In the instantiation, there may be a need to align the ciphertext space with the output space of the deterministic scrambler, in which case a standard collision-resistant hash function can be used.

### 6.2 Extended Boyen-Waters Scheme

Given a sound and secure deterministic scrambler scheme ($\mathsf{DS.init}, \mathsf{DS.enc}$), the following scheme is achieved by extending Boyen-Waters scheme [14] in two steps. The first step is to replace the identity *id* in all algorithms with $\mathsf{DS.enc}(id, param)$. This step guarantees the forward-secure function privacy property, without affecting the IND-CPA and anonymity properties. The second step is to employ composite-order bilinear groups, similar to the Arriaga-Tang-Ryan IBE Scheme described in Section 5.2. This step guarantees that the elements in the secret key can be randomized by randomly-chosen elements from a subgroup.

| $\mathsf{Setup}(\lambda)$ | $\mathsf{Extract}(Msk, id)$ |
|---|---|
| $\Gamma = (p, q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q) = \mathcal{G}_C(\lambda)$<br>$\Gamma^* = (n = pq, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_p, g_q)$<br>$x, t_1, t_2, t_3, t_4 \overset{\$}{\leftarrow} \mathbb{Z}_n$<br>$\Omega = \hat{e}(g_p, g_p)^{xt_1t_2}, g_0, g_1 \overset{\$}{\leftarrow} \mathbb{G}_p$<br>$v_1 = g_p^{t_1}, v_2 = g_p^{t_2}, v_3 = g_p^{t_3}, v_4 = g_p^{t_4}$<br>$\mathcal{M} = \mathbb{G}_T, \mathcal{I} = \mathbb{Z}_n, param \overset{\$}{\leftarrow} \mathsf{DS.init}(\lambda)$<br>$Msk = (x, t_1, t_2, t_3, t_4)$<br>$params = (\Gamma^*, g_0, g_1, \Omega, v_1, v_2, v_3, v_4, \mathcal{M}, \mathcal{I}, param)$ | $r_1, r_2 \overset{\$}{\leftarrow} \mathbb{Z}_n,$<br>$x_0, x_1, x_2, x_3, x_4 \overset{\$}{\leftarrow} \mathbb{G}_q$<br>$d_0 = x_0 \cdot g_p^{r_1t_1t_2+r_2t_3t_4}$<br>$d_1 = x_1 \cdot g_p^{-xt_2} \cdot (g_0 g_1^{\mathsf{DS.enc}(id,param)})^{-r_1t_2}$<br>$d_2 = x_2 \cdot g_p^{-xt_1} \cdot (g_0 g_1^{\mathsf{DS.enc}(id,param)})^{-r_1t_1}$<br>$d_3 = x_3 \cdot (g_0 g_1^{\mathsf{DS.enc}(id,param)})^{-r_2t_4}$<br>$d_4 = x_4 \cdot (g_0 g_1^{\mathsf{DS.enc}(id,param)})^{-r_2t_3}$<br>$sk_{id} = (d_0, d_1, d_2, d_3, d_4)$ |
| $\mathsf{Enc}(m, id)$ | $\mathsf{Dec}(C, sk_{id})$ |
| $s, s_1, s_2 \overset{\$}{\leftarrow} \mathbb{Z}_n$<br>$\hat{c} = \Omega^s m, c_0 = (g_0 g_1^{\mathsf{DS.enc}(id,param)})^s, c_1 = v_1^{s-s_1}$<br>$c_2 = v_2^{s_1}, c_3 = v_3^{s-s_2}, c_4 = v_4^{s_2}$<br>$C = (\hat{c}, c_0, c_1, c_2, c_3, c_4)$ | $e_0 = \hat{e}(c_0, d_0), e_1 = \hat{e}(c_1, d_1)$<br>$e_2 = \hat{e}(c_2, d_2), e_3 = \hat{e}(c_3, d_3)$<br>$e_4 = \hat{e}(c_4, d_4)$<br>$m = \hat{c} \cdot e_0 \cdot e_1 \cdot e_2 \cdot e_3 \cdot e_4$ |

Based on the fact that Boyen-Waters IBE is both IND-CPA and anonymous based on the DLIN assumption, it is clear that the extended scheme achieves the same security properties given the deterministic scrambler scheme is sound. Next, we show that the scheme achieves the other two properties. The proofs appear in Appendix IV.

**Theorem 5.** *The extended scheme achieves the forward-secure function privacy property, if the employed deterministic scrambler scheme is secure.*

**Theorem 6.** *The extended scheme achieves the forward-secure key unlinkability property, if the employed deterministic scrambler scheme is secure.*

## 7 Concluding Remarks

In this paper, we have defined two forward-secure properties for PEKS and IBE respectively. Moreover, we have analysed several existing PEKS and IBE schemes, and extended the Boyen-Waters anonymous IBE scheme by using a new building block (i.e. deterministic scrambler). Our analysis shows that it is relatively easy to achieve our properties against RO-non-adaptive attackers. However, it is not an easy task to construct secure schemes against adaptive attackers. In fact, the task seems to be very difficult if we want to use standard hardness assumptions and get rid of random oracle model. Our work has motivated many interesting future directions, and some of them are listed below.

- The "extract-combine-augment" approach from [10, 11] is a very elegant approach to achieve function privacy without forward security. It remains open whether the schemes from [10, 11] achieve key unlinkability or even forward-secure key unlinkability. Moreover, as pointed out earlier, this approach introduces some algebraic structure into ciphertexts, which makes the schemes not forward-secure with respect to function privacy. It is an interesting task to augment these schemes and enhance the approach to achieve our forward-secure properties.
- As to the concept of deterministic scrambler, we only know one method to instantiate it, i.e. using the adaptive secure deterministic encryption scheme by Raghunathan, Segev and Vadhan [26]. Unfortunately, the scheme is only secure in the random oracle model, and it seems difficult to have a secure scheme in the standard model as pointed by Wichs [29]. It is a very interesting task to see how to instantiate a deterministic scrambler scheme in the standard model.
- Both PEKS and IBE are special types of functional encryption [12]. Hence, the concept of forward security is also valuable for other functional encryption schemes, including other PEKS variants and searchable encryption schemes in the symmetric-key setting. As shown in [2, 12], proposing appropriate security definitions for these schemes may be very tricky due to the more complex functionalities and the inequality of indistinguishability and simulation based approaches. This is a big open research area for the future.
- The forward-secure properties for IBE have been motivated by constructing forward-secure PEKS schemes. It is interesting to investigate their implications in other applications of IBE, and study their relationships with other special properties such as those from [16, 30].

## Acknowledgement.

The author would like to thank Afonso Arriaga and Vincenzo Iovino for their helpful discussions.

## 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. *J. Cryptol.*, 21(3):350–391, 2008.
2. S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, and A. Sahai. Function private functional encryption and property preserving encryption : New definitions and positive results. http://eprint.iacr.org/2013/744, 2013.
3. A. Arriaga, Q. Tang, and P. Ryan. Trapdoor privacy in asymmetric searchable encryption schemes. http://eprint.iacr.org/2013/330.pdf, 2013.
4. A. Arriaga, Q. Tang, and P. Ryan. Trapdoor privacy in asymmetric searchable encryption schemes. In D. Pointcheval and D. Vergnaud, editors, *Progress in Cryptology – AFRICACRYPT 2014*, volume 8469 of *LNCS*, pages 31–50. Springer, 2014.
5. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In A. Menezes, editor, *Advances in cryptology — CRYPTO 2007*, volume 4622 of *LNCS*, pages 535–552. Springer, 2007.
6. M. Bellare and S. K. Miner. A forward-secure digital signature scheme. In M. J. Wiener, editor, *Advances in Cryptology — CRYPTO 1999*, volume 1666 of *LNCS*, pages 431–448. Springer, 1999.
7. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In M. Franklin, editor, *Advances in Cryptology — CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55, 2004.
8. 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.
9. 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.
10. D. Boneh, A. Raghunathan, and G. Segev. Function-private identity-based encryption: Hiding the function in functional encryption. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology — CRYPTO 2013*, volume 8043 of *LNCS*, pages 461–478. Springer, 2013.
11. D. Boneh, A. Raghunathan, and G. Segev. Function-private identity-based encryption: Hiding the function in functional encryption. http://eprint.iacr.org/2013/283.pdf, 2013.
12. D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In Y. Ishai, editor, *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, 2011.
13. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In S. P. Vadhan, editor, *Proceedings of the 4th conference on Theory of cryptography*, volume 4392 of *LNCS*, pages 535–554. Springer, 2007.
14. X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In C. Dwork, editor, *Advances in Cryptology — CRYPTO 2006*, volume 4117 of *LNCS*, pages 290–307. Springer, 2006.
15. R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In B. S. Kaliski Jr., editor, *Advances in Cryptology — CRYPTO 1997*, volume 1294 of *Lecture Notes in Computer Science*, pages 455–469. Springer, 1997.
16. S. M. Chow. Removing escrow from identity-based encryption. In S. Jarecki and G. Tsudik, editors, *Public Key Cryptography – PKC 2009, 12th International Conference on Practice and Theory in Public Key Cryptography*, volume 5443 of *LNCS*, pages 256–276. Springer, 2009.
17. I. Damgård, C. Hazay, and A. Zottarel. Short paper on the generic hardness of DDH-II. http://cs.au.dk/ angela/Hardness.pdf, Accessed in May, 2014.
18. W. Diffie, P. C. Oorschot, and M. J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2(2):107–125, 1992.
19. V. Goyal, A. O'Neill, and V. Rao. Correlated-input secure hash functions. In Y. Ishai, editor, *Theory of Cryptography — 8th Theory of Cryptography Conference, TCC 2011*, volume 6597 of *Lecture Notes in Computer Science*, pages 182–200. Springer, 2011.
20. C. Günther. An identity-based key-exchange protocol. In J. Quisquater and J. Vandewalle, editors, *Advances in Cryptology — EUROCRYPT 1989*, volume 434 of *Lecture Notes in Computer Science*, pages 29–37. Springer, 1990.
21. 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, editor, *Pairing-Based Cryptography – Pairing 2007*, volume 4575 of *LNCS*, pages 2–22. Springer, 2007.
22. V. Iovino and G. Persiano. Hidden-vector encryption with groups of prime order. In S. D. Galbraith and K. G. Paterson, editors, *Pairing-Based Cryptography – Pairing 2008*, volume 5209 of *LNCS*, pages 75–88. Springer, 2008.

23. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. P. Smart, editor, *Advances in cryptology — EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 146–162. Springer, 2008.
24. A. B. Lewko and B. Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In D. Micciancio, editor, *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 455–479. Springer, 2010.
25. M. Nishioka. Perfect keyword privacy in PEKS systems. In T. Takagi, G. Wang, Z. Qin, S. Jiang, and Y. Yu, editors, *Provable Security - 6th International Conference, ProvSec 2012*, volume 7496 of *Lecture Notes in Computer Science*, pages 175–192. Springer, 2012.
26. A. Raghunathan, G. Segev, and S. P. Vadhan. Deterministic public-key encryption for adaptively chosen plaintext distributions. In T. Johansson and P. Q. Nguyen, editors, *Advances in Cryptology — EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 93–110. Springer, 2013.
27. E. Shi, J. Bethencourt, T-H. H. Chan, D. Song, and A. Perrig. Multi-dimensional range query over encrypted data. In *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, pages 350–364. IEEE Computer Society, 2007.
28. L. Trevisan and S. Vadhan. Extracting randomness from samplable distributions. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 32–42. IEEE Computer Society, 2000.
29. D. Wichs. Barriers in cryptography with weak, correlated and leaky sources. In R. D. Kleinberg, editor, *Innovations in Theoretical Computer Science, ITCS 2013*, pages 111–126. ACM, 2013.
30. D. Yao, N. Fazio, Y. Dodis, and A. Lysyanskaya. Id-based encryption for complex hierarchies with applications to forward security and broadcast encryption. In V. Atluri, B. Pfitzmann, and P. D. McDaniel, editors, *Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004*, pages 354–363. ACM, 2004.

## Appendix I:

**Proof of Lemma 1**. Let $Y_0, Y_1$ be defined as follows.

$$Y_0 = (\Gamma^*,\ g_p^{b_1} \cdot g_q^{s_2},\ g_p^{a_1 b_1} \cdot g_q^{s_3}),\ \ Y_1 = (\Gamma^*,\ g_p^{b_1} \cdot g_q^{s_2},\ g_p^{r} \cdot g_q^{s_3})$$

It is clear that, given $Y_b$, the attacker's advantage in telling $b$ is not larger than $\epsilon$. Based on the standard hybrid argument, it is clear the lemma holds for the first scenario.

For the second scenario, we carry out the proof by an induction on $L$. Referring to the $X_0, X_1$ in Fig. 4, let $\alpha_0 = \Pr[\mathcal{A}(\Gamma^*, X_0)] = 1$ and $\alpha_1 = \Pr[\mathcal{A}(\Gamma^*, X_1)] = 1$. It is clear that $|\alpha_0 - \alpha_1| = 2\epsilon$. First, we consider the case when $L = 2$.

$$\alpha_1^{(2)} = \Pr[\mathcal{A}(\Gamma^*,\ g_p^{b_1} \cdot g_q^{s_1},\ g_p^{a_1 b_1} \cdot g_q^{t_1},\ g_p^{b_2} \cdot g_q^{s_2}, g_p^{a_1 b_2} \cdot g_q^{t_2}) = 1]$$

$$\alpha_2^{(2)} = \Pr[\mathcal{A}(\Gamma^*,\ g_p^{b_1} \cdot g_q^{s_1},\ g_p^{r_1} \cdot g_q^{t_1}, g_p^{b_2} \cdot g_q^{s_2},\ g_p^{r_2} \cdot g_q^{t_2}) = 1]$$

Let $\beta^{(2)}$ be defined as follows.

$$\beta^{(2)} = \Pr[\mathcal{A}(\Gamma^*,\ g_p^{b_1} \cdot g_q^{s_1},\ g_p^{a_1 b_1} \cdot g_q^{t_1},\ g_p^{b_2} \cdot g_q^{s_2},\ g_p^{r} \cdot g_q^{t_2}) = 1]$$

It is straightforward to verify that $|\alpha_1^{(2)} - \beta^{(2)}| \le 2\epsilon$ and $|\alpha_2^{(2)} - \beta^{(2)}| \le 2\epsilon$ based on the Composite-DDH-II assumption. Therefore, we have $|\alpha_1^{(2)} - \alpha_2^{(2)}| \le 4 \cdot \epsilon$.

Suppose that $|\alpha_1^{(n)} - \alpha_2^{(n)}| \le 2n \cdot \epsilon$, we prove that $|\alpha_1^{(n+1)} - \alpha_2^{(n+1)}| \le 2(n+1) \cdot \epsilon$.

$$\alpha_1^{(n+1)} = \Pr[\mathcal{A}(\Gamma^*, g_p^{b_1} \cdot g_q^{s_1}, g_p^{a_1 b_1} \cdot g_q^{t_1}, \cdots, g_p^{b_n} \cdot g_q^{s_n}, g_p^{a_1 b_n} \cdot g_q^{t_n}, g_p^{b_{n+1}} \cdot g_q^{s_{n+1}}, g_p^{a_1 b_{n+1}} \cdot g_q^{t_{n+1}}) = 1]$$

$$\alpha_2^{(n+1)} = \Pr[\mathcal{A}(\Gamma^*, g_p^{b_1} \cdot g_q^{s_1}, g_p^{r_1} \cdot g_q^{t_1}, \cdots, g_p^{b_n} \cdot g_q^{t_n}, g_p^{r_n} \cdot g_q^{t_n}, g_p^{b_{n+1}} \cdot g_q^{t_{n+1}}, g_p^{r_{n+1}} \cdot g_q^{t_{n+1}}) = 1]$$

Let $\beta^{(n+1)}$ be defined as follows.

$$\beta^{(n+1)} = \Pr[\mathcal{A}(\Gamma^*, g_p^{b_1} \cdot g_q^{s_1}, g_p^{r_1} \cdot g_q^{t_1}, \cdots, g_p^{b_n} \cdot g_q^{t_n}, g_p^{r_n} \cdot g_q^{t_n}, g_p^{b_n \cdot r} \cdot g_q^{t_{n+1}}, g_p^{r_n \cdot r} \cdot g_q^{t_{n+1}}) = 1]$$

It is straightforward to verify that $|\alpha_2^{(n+1)} - \beta^{(n+1)}| \le 2\epsilon$ based on the Composite-DDH-II assumption. With respect to $\alpha_1^{(n+1)}$ and $\beta^{(n+1)}$, we observe that the last the last two terms $(g_p^{b_{n+1}} \cdot g_q^{s_{n+1}}, g_p^{a_1 b_{n+1}} \cdot g_q^{t_{n+1}})$ and $(g_p^{b_n \cdot r} \cdot g_q^{t_{n+1}}, g_p^{r_n \cdot r} \cdot g_q^{t_{n+1}})$ can be unanimously generated by their previous terms. Therefore, we have $|\alpha_1^{(n+1)} - \beta^{(n+1)}| = |\alpha_1^{(n)} - \alpha_2^{(n)}|$ based on the fact that $\alpha_2^{(n)}$ is identical to $\beta^{(n+1)}$ without $(g_p^{b_n \cdot r} \cdot g_q^{t_{n+1}}, g_p^{r_n \cdot r} \cdot g_q^{t_{n+1}})$. Now, we have $|\alpha_1^{(n+1)} - \alpha_2^{(n+1)}| \le 2\epsilon + |\alpha_1^{(n)} - \alpha_2^{(n)}| \le 2(n+1) \cdot \epsilon$.

To sum up, we have $|\alpha_1^{(L)} - \alpha_2^{(L)}| \le 2L \cdot \epsilon$, so that $|\Pr[b' = b] - \frac{1}{2}| \le L \cdot \epsilon$ in the attack game defined in Fig. 5. The lemma now follows. □

Appendix II:

**Proof of Theorem 1.** Referring to the attack game for forward-secure function privacy, as shown in Fig. 9, the attacker is given the trapdoor and ciphertext vectors $\boldsymbol{T}_b, \boldsymbol{C}_b$ to guess $b$. In addition, the attacker can query more trapdoors and ciphertexts for the keywords behind $\boldsymbol{T}_b, \boldsymbol{C}_b$.

To simplify the proof, we assume that the challenger sets the challenges to the following form:

$$\boxed{\text{b=0}}: \quad (\mathsf{H}(\boldsymbol{x}_0^{(1)}), \mathsf{H}(\boldsymbol{x}_0^{(2)}), \cdots, \mathsf{H}(\boldsymbol{x}_0^{(L)}))$$

$$\boxed{\text{b=1}}: \quad (\mathsf{H}(\boldsymbol{x}_1^{(1)}), \mathsf{H}(\boldsymbol{x}_1^{(2)}), \cdots, \mathsf{H}(\boldsymbol{x}_1^{(L)}))$$

It is easy to see that, with the challenge, the attacker can simulate all the information it is allowed to receive. In fact, the attacker receives more information than in a faithful attack game. With respect to the simplified challenges, it is clear that they are indistinguishable unless the attacker has queried $\boldsymbol{x}_b^{(j)}$ for some $1 \le j \le L$ to the random oracle $\mathsf{H}$. Suppose the attacker issues $t$ queries to the random oracle, this event occurs with the probability $\frac{t \cdot L}{2^\lambda}$.

As a result, the attacker's advantage is bounded by $\frac{t \cdot L}{2^\lambda}$ in the faithful attack game, and the theorem follows. □

**Proof of Theorem 2**. Referring to the attack game for forward-secure trapdoor unlinkability, as shown in Fig. 8, the attacker is given the trapdoor vector $\boldsymbol{T}_b$ to guess $b$. The $\mathsf{TrapGen}$ oracle is not helpful to the attacker, since it is trivial to rerandomize any give trapdoor to obtain a new trapdoor for the same keyword. The received challenges by the attacker are in the following form. Let $\boldsymbol{x}_0 = (\boldsymbol{x}_0^{(1)}, \boldsymbol{x}_0^{(2)}, \cdots, \boldsymbol{x}_0^{(L)}) \overset{\mathbb{D}_0}{\leftarrow} \mathcal{W}^L$ and $\boldsymbol{x}_1 = (\boldsymbol{x}_1^{(1)}, \boldsymbol{x}_1^{(2)}, \cdots, \boldsymbol{x}_1^{(L)}) \overset{\mathbb{D}_1}{\leftarrow} \mathcal{W}^L$.

$$\boxed{\text{b=0}}: \quad ((g_p^{r_1^{(1)}} \cdot g_w'^{(1)}, \mathsf{H}(\boldsymbol{x}_0^{(1)})^{r_1^{(1)}} \cdot g_w''^{(1)}), \cdots, (g_p^{r_1^{(L)}} \cdot g_w'^{(L)}, \mathsf{H}(\boldsymbol{x}_0^{(L)})^{r_1^{(L)}} \cdot g_w''^{(L)}))$$

$$\boxed{\text{b=1}}: \quad ((g_p^{r_1^{(1)}} \cdot g_w'^{(1)}, \mathsf{H}(\boldsymbol{x}_1^{(1)})^{r_1^{(1)}} \cdot g_w''^{(1)}), \cdots, (g_p^{r_1^{(L)}} \cdot g_w'^{(L)}, \mathsf{H}(\boldsymbol{x}_1^{(L)})^{r_1^{(L)}} \cdot g_w''^{(L)}))$$

The rest of the proof is identical to the proof of Theorem 4, where we show that $|\Pr[b' = b] - \frac{1}{2}|$ is negligible based on the Weak Composite-DDH assumption in the random oracle model. The theorem follows. □

Appendix III:

**Proof of Theorem 4**. Recall that in the attack game, shown in Fig. 14, the attacker is given *Msk* in Step 4. In our proof, we simply give *Msk* to the attacker in Step 2, so that there is no need for the attacker to submit any oracle queries (except for the random oracle $\mathsf{H}$) to the challenger anymore. To further simplify our description, we assume the secret key for an identity *id* in the challenge is in the following form.

$$x, y \overset{\$}{\leftarrow} \mathbb{G}_q,\ sk_{id} = (d_0, d_1), d_0 = x \cdot g_p^r, d_1 = y \cdot \mathsf{H}(id)^r$$

It is straightforward to verify that the attacker can extend the simplified secret key to its original form, and we skip the details here. The attack game can be simplified as follows.

```
1. $(Msk, params) \overset{\$}{\leftarrow} \mathsf{Setup}(\lambda)$
2. $(\mathbb{D}_0, \mathbb{D}_1, L, state) \overset{\$}{\leftarrow} \mathcal{A}(params, Msk)$
3. $b \in_R \{0,1\},\ \boldsymbol{id}_b \leftarrow \mathbb{D}_b,\ \boldsymbol{sk}_b = \mathsf{Extract}(\boldsymbol{id}_b, Msk)$
4. $b' \overset{\$}{\leftarrow} \mathcal{A}(state, \boldsymbol{sk}_b)$
```

After the simplification, the received challenges by the attacker are in the following form.

$$\boxed{\text{b=0}}: \quad ((x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_0^{(1)})^{r_1}),\ \cdots,\ (x_L \cdot g_p^{r_L},\ y_L \cdot \mathsf{H}(\boldsymbol{id}_0^{(L)})^{r_L}))$$

$$\boxed{\text{b=1}}:\quad ((x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_1^{(1)})^{r_1}),\ \cdots,\ (x_L \cdot g_p^{r_L},\ y_L \cdot \mathsf{H}(\boldsymbol{id}_1^{(L)})^{r_L}))$$

Next, we carry out the proof by an induction on $L$.

Case $L = 1$. In the faithful game, the challenge is $sk_{\boldsymbol{id}_b} = sk_{\boldsymbol{id}_b^{(1)}}$, where

$$\boldsymbol{id}_0 = \boldsymbol{id}_0^{(1)} \xleftarrow{\mathbb{D}_0} \mathcal{I},\ sk_{\boldsymbol{id}_0^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_0^{(1)})^{r_1})$$

$$\boldsymbol{id}_1 = \boldsymbol{id}_1^{(1)} \xleftarrow{\mathbb{D}_1} \mathcal{I},\ sk_{\boldsymbol{id}_1^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_1^{(1)})^{r_1})$$

Let the attacker's advantage be $\epsilon_1$. Consider a new game, where the challenge $sk_{\boldsymbol{id}_b}$ is generated as follows.

$$\alpha \xleftarrow{\$} \mathbb{G}_p,\ sk_{\boldsymbol{id}_0^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \alpha^{r_1})$$

$$\alpha \xleftarrow{\$} \mathbb{G}_p,\ sk_{\boldsymbol{id}_1^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \alpha^{r_1})$$

Suppose the attacker issues $h$ queries to the random oracle $\mathsf{H}$ in the game. The new game is identical to the original one with the probability $1 - \frac{h}{2^\lambda}$, where $\frac{h}{2^\lambda}$ is the probability that the attacker has queried one of the identities in $\boldsymbol{id}_0$ and $\boldsymbol{id}_1$ to the random oracle. It is clear that the attacker's advantage is 0 when the games are identical. As a result, $\epsilon_1 \leq \frac{h}{2^\lambda}$.

Case $L = 2$. In the faithful game, the challenge is $sk_{\boldsymbol{id}_b} = (sk_{\boldsymbol{id}_b^{(1)}}, sk_{\boldsymbol{id}_b^{(2)}})$, where

$$\boldsymbol{id}_0 = (\boldsymbol{id}_0^{(1)}, \boldsymbol{id}_0^{(2)}) \xleftarrow{\mathbb{D}_0} (\mathcal{I}, \mathcal{I}),\ sk_{\boldsymbol{id}_0^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_0^{(1)})^{r_1}),\ sk_{\boldsymbol{id}_0^{(2)}} = (x_2 \cdot g_p^{r_2},\ y_2 \cdot \mathsf{H}(\boldsymbol{id}_0^{(2)})^{r_2})$$

$$\boldsymbol{id}_1 = (\boldsymbol{id}_1^{(1)}, \boldsymbol{id}_1^{(2)}) \xleftarrow{\mathbb{D}_1} (\mathcal{I}, \mathcal{I}),\ sk_{\boldsymbol{id}_1^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_1^{(1)})^{r_1}),\ sk_{\boldsymbol{id}_1^{(2)}} = (x_2 \cdot g_p^{r_2},\ y_2 \cdot \mathsf{H}(\boldsymbol{id}_1^{(2)})^{r_2})$$

Let the attacker's advantage be $\epsilon_2$. Consider a new game, which is faithful except for the challenge generation.

- If $\boldsymbol{id}_0^{(1)} \neq \boldsymbol{id}_0^{(2)}$, the challenge $sk_{\boldsymbol{id}_0}$ is generated as follows.

  $$\alpha, \beta \xleftarrow{\$} \mathbb{G}_p,\ sk_{\boldsymbol{id}_0^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \alpha^{r_1}),\ sk_{\boldsymbol{id}_0^{(2)}} = (x_2 \cdot g_p^{r_2},\ y_2 \cdot \beta^{r_2})$$

  Otherwise, the challenge $sk_{\boldsymbol{id}_0}$ is generated as follows.

  $$\alpha \xleftarrow{\$} \mathbb{G}_p,\ sk_{\boldsymbol{id}_0^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \alpha^{r_1}),\ sk_{\boldsymbol{id}_0^{(2)}} = (x_2 \cdot g_p^{r_2},\ y_2 \cdot \alpha^{r_2})$$

- The challenge $sk_{\boldsymbol{id}_1}$ is always generated as follows.

  $$\alpha, \beta \xleftarrow{\$} \mathbb{G}_p,\ sk_{\boldsymbol{id}_1^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \alpha^{r_1}),\ sk_{\boldsymbol{id}_1^{(2)}} = (x_2 \cdot g_p^{r_2},\ y_2 \cdot \beta^{r_2})$$

Suppose the attacker issues $h$ queries to the random oracle $\mathsf{H}$ in the game. The new game is identical to the original one with the probability $1 - \frac{2h}{2^\lambda}$, where $\frac{2h}{2^\lambda}$ is the probability that the attacker has queried one of the identities in $\boldsymbol{id}_0$ and $\boldsymbol{id}_1$ to the random oracle. When the games are identical, we can compute the attacker's advantage by considering two cases.

- One case is $\boldsymbol{id}_0^{(1)} \neq \boldsymbol{id}_0^{(2)}$. Let $p_1 = \Pr[\boldsymbol{id}_0^{(1)} \neq \boldsymbol{id}_0^{(2)}]$ according to $\mathbb{D}_0$. In this case, the attacker's advantage is 0.
- The other case is $\boldsymbol{id}_0^{(1)} = \boldsymbol{id}_0^{(2)}$. Let $p_2 = \Pr[\boldsymbol{id}_0^{(1)} = \boldsymbol{id}_0^{(2)}]$ according to $\mathbb{D}_0$. In this case, the attacker's advantage is exactly $Adv_{wcddh}$, which is the attacker's advantage in the Weak Composite-DDH assumption.

Combining the two cases, the attacker's overall advantage is $p_2 \cdot Adv_{wcddh}$ when the new game is identical to the original one. As a result, $\epsilon_2 \leq \frac{2h}{2^\lambda} + Adv_{wcddh}$.

Case $L = 3$. In the faithful game, the challenge is $sk_{\boldsymbol{id}_b} = (sk_{\boldsymbol{id}_b^{(1)}}, sk_{\boldsymbol{id}_b^{(2)}}, sk_{\boldsymbol{id}_b^{(3)}})$, where

$$\boldsymbol{id}_0 = (\boldsymbol{id}_0^{(1)}, \boldsymbol{id}_0^{(2)}, \boldsymbol{id}_0^{(3)}) \xleftarrow{\mathbb{D}_0} (\mathcal{I}, \mathcal{I}, \mathcal{I}),\ sk_{\boldsymbol{id}_0^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_0^{(1)})^{r_1})$$

$$sk_{\boldsymbol{id}_0^{(2)}} = (x_2 \cdot g_p^{r_2},\ y_2 \cdot \mathsf{H}(\boldsymbol{id}_0^{(2)})^{r_2}),\ sk_{\boldsymbol{id}_0^{(3)}} = (x_3 \cdot g_p^{r_3},\ y_3 \cdot \mathsf{H}(\boldsymbol{id}_0^{(3)})^{r_3})$$

$$\boldsymbol{id}_1 = (\boldsymbol{id}_1^{(1)}, \boldsymbol{id}_1^{(2)}, \boldsymbol{id}_1^{(3)}) \xleftarrow{\mathbb{D}_1} (\mathcal{I}, \mathcal{I}, \mathcal{I}),\ sk_{\boldsymbol{id}_1^{(1)}} = (x_1 \cdot g_p^{r_1},\ y_1 \cdot \mathsf{H}(\boldsymbol{id}_1^{(1)})^{r_1})$$

$$sk_{\boldsymbol{id}_1^{(2)}} = (x_2 \cdot g_p^{r_2},\ y_2 \cdot \mathsf{H}(\boldsymbol{id}_1^{(2)})^{r_2}),\ sk_{\boldsymbol{id}_1^{(3)}} = (x_3 \cdot g_p^{r_3},\ y_3 \cdot \mathsf{H}(\boldsymbol{id}_1^{(3)})^{r_3})$$

Let the attacker’s advantage be $\epsilon_3$

- The other case is $\boldsymbol{id}_0^{(i)} = \boldsymbol{id}_0^{(j)}$ for some $1 \leq i \neq j \leq n+1$. Let $p_2$ be the probability of this case according to $\mathbb{D}_0$. In this case, the attacker's advantage is $|\epsilon_n - \frac{nh}{2^\lambda}| + Adv_{wcddh}$ for the same reason as in computing $q_1$ in the case of $L = 3$.

Combining both cases, the attacker's advantage is $p_2(|\epsilon_n - \frac{nh}{2^\lambda}| + Adv_{wcddh}) \leq |\epsilon_n - \frac{nh}{2^\lambda}| + Adv_{wcddh}$ when the new game is identical to the original one. As a result, the attacker's advantage in the original game is

$$\epsilon_{n+1} \leq |\epsilon_n - \frac{nh}{2^\lambda}| + Adv_{wcddh} + \frac{(n+1) \cdot h}{2^\lambda}. \tag{4}$$

Conclusion. Based on the inequalities (3) and (4), we have $\epsilon_L \leq +(L-1) \cdot Adv_{wcddh} + \frac{L \cdot h}{2^\lambda}$. The theorem is proven. □

Appendix IV:

**Proof of Theorem 5**. Note the fact that the requirements on the distributions $\mathbb{D}_0, \mathbb{D}_1$ in the forward-secure function privacy attack game (Fig. 15) and deterministic scrambler attack game (Fig. 16) are the same. Given an attacker for the forward-secure function privacy property, we can construct an attacker for the deterministic scrambler's security property with the same advantage. As a result, the theorem naturally follows. □

**Proof of Theorem 6**. For simplicity, suppose *Msk* is public. Then, the private key $sk_{id}$ for an identity *id* can be simplified to the following form.

$$x, y \xleftarrow{\$} \mathbb{G}_q,\ r \xleftarrow{\$} \mathbb{Z}_n,\ sk_{id} = (\alpha, \beta), \alpha = x \cdot g_p^r, \beta = y \cdot (g_0 g_1^{\mathsf{DS.enc}(id)})^r.$$

It is straightforward to verify that, given a key in the above form, an attacker can patch it to a key defined in the original form.

With this simplification, referring to the attack game in Fig. 14, the challenge is in the following form, where $x_1, \cdots, x_L, y_1, \cdots, y_L \xleftarrow{\$} \mathbb{G}_q$ and $s_1, \cdots, s_L, r_1, \cdots, r_L \xleftarrow{\$} \mathbb{Z}_n$.

$$\boxed{\text{b=0}}:\ (x_1 \cdot g_p^{r_1}, y_1 \cdot (g_0 g_1^{\mathsf{DS.enc}(\boldsymbol{id}_0^{(1)}, param)})^{r_1}; \cdots; x_L \cdot g_p^{r_L}, y_L \cdot (g_0 g_1^{\mathsf{DS.enc}(\boldsymbol{id}_0^{(L)}, param)})^{r_L})$$

$$\boxed{\text{b=1}}:\ (x_1 \cdot g_p^{r_1}, y_1 \cdot (g_0 g_1^{\mathsf{DS.enc}(s_1, param)})^{r_1}; \cdots; x_L \cdot g_p^{r_L}, y_L \cdot (g_0 g_1^{\mathsf{DS.enc}(s_L, param)})^{r_L})$$

Due to the soundness property of the deterministic scrambler scheme, the DS.enc algorithm is an injective function with overwhelming probability. Hence, it does not change the min-entropy of the input identities. As a result, we can conclude that $g_0 g_1^{\mathsf{DS.enc}(\boldsymbol{id}_0^{(i)}, param)}$ for every $1 \leq i \leq L$ has min-entropy $\lambda$. On the other hand, $g_0 g_1^{\mathsf{DS.enc}(s_i, param)}$ for every $1 \leq i \leq L$ is uniformly distributed. Based on these facts, $|\Pr[b' = b] - \frac{1}{2}|$ is negligible if the correlated Composite-DDH-II assumption holds. The theorem follows. □