Public-Key Encryption with Cluster-Chain-based Keyword Search

Peng Xu, Hai Jin, Wei Wang, Deqing Zou
Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology
Huazhong University of Science and Technology, China
xupeng@mail.hust.edu.cn

January 6, 2012

Abstract

It is widely acknowledged that the keyword search performance and the privacy of keywords are equally important for keyword searchable ciphertexts. To our knowledge, no public-key-encryption-based work in literature can accelerate the keyword search, while preserving semantic security of keywords under chosen keyword attacks. In this paper, we propose public-key encryption with cluster-chain-based keyword search (PCCS), which is an innovation of public-key encryption with keyword search (PEKS). PCCS not only has much more efficient keyword search, but also preserves the equal SS-CKA security with PEKS from computational bilinear Diffie-Hellman (CBDH) assumption. With such advantages, PCCS just slightly increases the size of public parameter and the space complexity of keyword searchable ciphertexts.
Keyword: keyword search, public-key encryption with keyword search, semantic security of keywords under chosen keyword attacks, computational bilinear Diffie-Hellman assumption

1 Introduction

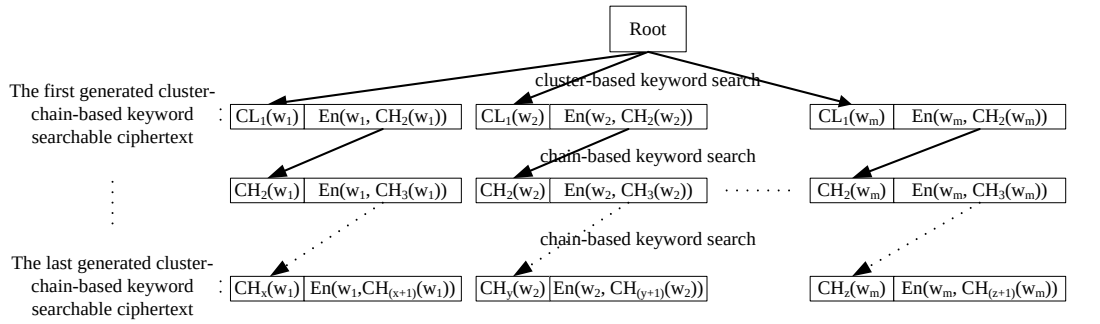
Generally speaking, searchability over encrypted keywords is an efficient algorithm to find out the queried keywords while preserving their privacy. This feature has been widely used to securely search keywords of user's sensitive datum, which are stored in a third party. In mostly related works, we can easily observe that a strong privacy of keywords usually induces a low efficient keyword search. But we have to admit the fact that the strong privacy is as important as the high efficiency. Therefore, we are interested in improving the efficiency of keyword search while preserving a strong privacy of keywords (semantic security of keywords under chosen keyword attacks).

Public-Key Encryption with Keyword Search (PEKS), proposed by Boneh, Crescenzo, Ostrovsky and Persiano [1], is the first keyword searchable encryption based on the randomized public-key encryption. Compared with previously similar works based on symmetric encryption like [2, 3, 4, 5], PEKS can be more convenient in practice. In the application of PEKS, any sender (who generates keyword searchable ciphertexts) can straightly generates keyword searchable ciphertexts with the public key of PEKS. It is free for them to join in and exit the application. Contrarily, in the application of previous works, a secret should be initially shared between each sender and the receiver (who issues the query of keyword and receives the response). To resist inner attack, this

initialization process will be more complex. In addition, PEKS would be more secure than mostly previous works. The previous works based on deterministic symmetric-key encryption can not meet the security of PEKS [36]. Even some of them like [2] suffer from the statistic analysis attack. The other previous works based on randomized symmetric-key encryption like [5] maybe have the similar security with PEKS in the best case. But they can not be more secure than PEKS.

However, it is widely acknowledged that the high security of PEKS is disadvantageous to its keyword search performance. In PEKS, keyword searchable ciphertexts have the uniform distribution at least in the view of computationally limited adversary. So without a correct keyword search trapdoor, nobody can distinguish whether any two keyword searchable ciphertexts contain the same keyword. To respond a query (when receiving a keyword search trapdoor), PEKS sequentially tests all keyword searchable ciphertexts that whether they contain the queried keyword. Furthermore, each test has the same complexity. Therefore, the keyword search complexity of PEKS is linear in the sum of keyword searchable ciphertexts. when the sum is too large, the keyword search perform of PEKS would be inefficient in practice [5, 36]. But in some previous works like [3, 20, 5], their keyword search are much efficient than PEKS. So the low keyword search efficiency is the main flaw of PEKS.

1.1 Our Contributions



Note: All CL(w_i) and CH(w_i) are randomized values. En(w_i, CH(w_i)) is a public-key encryption with the public-key w_i and the plaintext CH(w_i).

Figure 1: Framework of public-key encryption with cluster-chain-based keyword search

In this paper, we propose a novel method called *Public-Key Encryption with Cluster-Chain-based Keyword Search (PCCS). Our PCCS has much more efficient keyword search than PEKS, while preserving the equal security with PEKS* (that is, preserving semantic security of keywords under chosen keyword attacks (SS-CKA) [1, 6]). Let $\mathcal{W}$ be keyword space. Let T_{w_i} denote the sum of keyword searchable ciphertexts containing the keyword w_i. The sum of keyword searchable ciphertexts is $T = \sum_{w_i \in \mathcal{W}} T_{w_i}$. To respond a query of any keyword w_i, the search complexity of PEKS is linear in a much bigger value $\sum_{w_i \in \mathcal{W}} T_{w_i}$, but the search complexity of our PCCS is linear in T_{w_i}. In addition, our PCCS just slightly increases the size of public parameter and the space complexity of keyword searchable ciphertexts.

Framework of PCCS. In PCCS, each sender constructs a cluster-chain-based relationship for all his cluster-chain-based keyword searchable ciphertexts. Furthermore, this relationship is hidden at least in the view of computational limited adversary. Figure 1 shows a cluster-chain-based

relationship of a sender. To search the keyword W_i in Figure 1, firstly compute $CL_1(W_i)$ by taking as input *Root* and the keyword search trapdoor of W_i, and find out the first cluster-chain-based keyword searchable ciphertext $CL_1(W_i)||En(W_i, CH_2(W_i))$; Secondly, compute $CH_2(W_i)$ by decrypting $En(W_i, CH_2(W_i))$, and find out the second cluster-chain-based keyword searchable ciphertext $CH_2(W_i)||En(W_i, CH_3(W_i))$; Similarly, find out all cluster-chain-based keyword searchable ciphertexts of W_i. For all senders' cluster-chain-based relationships, their search processes are the same. So the search complexity of PCCS relies on the times of decryption operations. It is linear in T_{w_i}.

Public-Key Encryption with Cluster-based Keyword Search (PCS). PCS is the foundation of PCCS. Referring to Figure 1, it achieves three functions: 1. With a public parameter and a particular secret (randomly chosen by a sender), it can one-way compute *Root*; 2. With a keyword W_i and the particular secret, it can one-way compute the cluster-based keyword searchable ciphertext $CL_1(W_i)$; 3. With a randomized keyword search trapdoor of W_i and *Root*, it can find out the corresponding cluster-chain-based keyword searchable ciphertext by computing $CL_1(W_i)$. To realize PCS, we extend the traditional discrete logarithm (DL) assumption to group-based square matrix. In other words, it is hard to compute $\begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix}$ from $\begin{pmatrix} g^{r_{1,1}} & \cdots & g^{r_{1,m}} \\ \vdots & \ddots & \vdots \\ g^{r_{m,1}} & \cdots & g^{r_{m,m}} \end{pmatrix}$, where for all $i, j \in [1, m]$ $g^{r_{i,j}}$ are randomly chosen from a group $\mathbb{G}$ of prime order q. We propose the computation $\begin{pmatrix} g^{r_{1,1}} & \cdots & g^{r_{1,m}} \\ \vdots & \ddots & \vdots \\ g^{r_{m,1}} & \cdots & g^{r_{m,m}} \end{pmatrix} \cdot \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix}^{-1} = \begin{pmatrix} g & & 1 \\ & \ddots & \\ 1 & & g \end{pmatrix}$, which is the foundation to achieve the third function of PCS. In respect of the security of PCS, we define PCS's SS-CKA security in the random oracle (RO) model, and prove this security from computational blinear Diffie-Hellman (CBDH) assumption [7].

SS-CKA Secure PCCS From CBDH Assumption. PCCS can be realized by combining PCS with any anonymous identity-based encryption (IBE) scheme [7, 11, 12, 13, 14]. In 2001, Boneh and Franklin [7] proposed the most efficient anonymous IBE scheme BF01. Referring to Figure 1, BF01 is used to encrypt a random value when generating a cluster-chain-based keyword searchable ciphertext. Furthermore, the ciphertext has a chain-based relationship with its next ciphertext by taking the random value as the first part of the next ciphertext. In respect of the security of PCCS, we define PCCS's SS-CKA security in RO model, and prove this security from PCS's SS-CKA security and BF01's Anon-IND-ID-CPA security [7] (Anon-IND-ID-CPA security will be presented in Subsection 2.3). Furthermore, since BF01 is Anon-IND-ID-CPA secure from CBDH assumption, we can easily prove that PCCS is SS-CKA secure in the RO model from CBDH assumption.

Slight Increase in the Size of Public Parameter and the Space Complexity. Denote the size of keyword space by $|\mathcal{W}|$. The size of public parameter of PCCS mostly relies on the binary length of keyword space. It is linear in $(\log |\mathcal{W}|)^2$. In respect of the space complexity of keyword searchable ciphertexts, PEKS is linear in a function $f(|\mathcal{W}| \cdot N)$, where N denotes the sum of senders.

Compared with that, the major increase of PCCS relies on the space complexity of roots. It is linear in $N \cdot \log |\mathcal{W}|$. Therefore, relative to $f(|\mathcal{W}| \cdot N)$, the increase $N \cdot \log |\mathcal{W}|$ is slight.

SS-CKA Secure PCCS in the standard model. According to our proposed PCCS, it is easy to improve the SS-CKA security of PCCS in the standard model. By straightly canceling the random oracle in PCS, the improved PCS can be SS-CKA secure from decision bilinear Diffie-Hellman assumption [11] in the standard model. By combining the improved PCS with the anonymous IBE scheme, which is secure in the standard model [11, 12], the new PCCS scheme can be SS-CKA secure in the standard model. Due to the page limitation, this work will appear in another paper.

1.2 Discussions and Related Works

Keyword searchability over encrypted data recently has attracted much interest from scholars. To our knowledge, the current state of the art keyword search in the symmetric-key setting has much efficient keyword search, while preserving a strong security. But in the public-key setting, no work can accelerate the keyword search without weakening the privacy of keyword.

Symmetric-Key Encryption with Keyword Search (SEKS). In 2000, Song et al. [2] proposed the first practical search over symmetric-key encrypted data. But it is not proven to be a secure searchable encryption, and its search performance is linear in the size of document collection. These limitations are addressed by the works [3, 19, 20, 18, 21]. But they fail to prove that their works are semantically secure under adaptive adversary. To our knowledge, the current state of the art SEKS work was proposed by Curtmola et al. [5] in 2006. It has much efficient keyword search, while preserving the provably semantic security under adaptive adversary. But its size of keyword search trapdoor is linear in the size of document collection, which may be the main limitation of this work. In addition, works in [5, 23] extended SEKS to the multi-user environment. The work in [22] realized fuzzy keyword search. The work in [24] employed SEKS to realize the secure and searchable audit log.

Public-Key Encryption with Keyword Search (PEKS). After the first proposition of PEKS by Boneh et al. [1], Abdalla et al. [6] further complete the theory of PEKS. The works in [25, 39] remove the secure channel of PEKS respectively in the RO model and the standard model. The most existing works pay attention to the variety of keyword search, like conjunctive search [26, 27, 31, 30, 32, 25], range search [33, 34, 28], subset search [28], time-scope search [40, 6], similarity search [29] and so on. The works in [30, 41] extend PEKS to multi-user setting. The works in [42, 43] proposed the secure integration of public-key encrypted data and PEKS. Camenisch et al. [44] proposed the oblivious generation of keyword search trapdoor to keep the privacy of keyword from the trapdoor generator. By taking huge effort, PEKS recently is well studied, but all above works suffer from the same problem that their keyword search performance are at least linear in the sum of keyword searchable ciphertexts. To solve this problem, Bellare et al. [36] proposed the deterministic public-key encryption to realize the highly efficient keyword search. The works of Bellare et al. [37] and of Boldyreva et al. [38] improved its security in the standard model. But their high efficiency sacrifice the original security of PEKS. So accelerating the keyword search while preserving the original security of PEKS is still hard.

2 Preliminaries

Let $r \xleftarrow{\$} R$ denote randomly choosing r from the space R or outputting r by a randomized algorithm R.

2.1 Bilinear Map

Let $\mathbb{G}$ and $\mathbb{G}_t$ denote two multiplicative groups of prime order q. Let g be a generator of $\mathbb{G}$. Bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_t$ [15, 16] is an efficiently computable and non-degenerate function, with the bilinearity that $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$, where $\langle a, b\rangle \xleftarrow{\$} \mathbb{Z}_q^*$ and $\hat{e}(g, g)$ is a generator of $\mathbb{G}_t$. Let $BGen(1^k)$ be an efficient bilinear map generator that takes as input a security parameter 1^k and randomized outputs $\langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}\rangle$.

2.2 CBDH Assumption

Based on an instance $\langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}\rangle \xleftarrow{\$} BGen(1^k)$, we define the advantage of a probabilistic polynomial time (PPT) adversary A to solve CBDH problem as

$$Adv_A^{CBDH}(1^k) = \Pr[A(q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, g^a, g^b, g^c) = \hat{e}(g, g)^{abc}] \text{ where } \langle g^a, g^b, g^c\rangle \xleftarrow{\$} \mathbb{G}^3$$

Definition 1 (CBDH Assumption) *We say that CBDH assumption holds in $BG(1^k)$ if no PPT adversary has a non-negligible advantage to solve CBDH problem.*

2.3 Anonymous IBE scheme BF01

BF01 [7] consists of the following four algorithms: *Setup*, *Extract*, *Encrypt*, *Decrypt*.

- $Setup(1^k)$: Given a security parameter 1^k, compute $\langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}\rangle \xleftarrow{\$} BGen(1^k)$; Pick $s \xleftarrow{\$} \mathbb{Z}_q^*$ and set $P_{pub} = g^s$; Choose two cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{G}$ and $H_2 : \mathbb{G}_t \to \{0,1\}^n$. The plaintext space is $\mathcal{M} = \{0,1\}^n$. The ciphertext space is $\mathcal{C} = \mathbb{G} \times \{0,1\}^n$. The public parameter is $BF01.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, H_1, H_2\rangle$. The private parameter is s.
- $Extract(\mathrm{ID}, s)$: Given an identity $\mathrm{ID} \in \{0,1\}^*$, generate its private key $d_{\mathrm{ID}} = H_1(\mathrm{ID})^s$.
- $Encrypt(BF01.Pub, \mathrm{ID}, M)$: To encrypt the plaintext $M \in \mathcal{M}$ with the public-key ID, choose $r \xleftarrow{\$} \mathbb{Z}_q^*$ and set the ciphertext to be

$$C = \langle g^r, H_2(\hat{e}(H_1(\mathrm{ID}), P_{Pub})^r) \oplus M\rangle$$

- $Decrypt(d_{\mathrm{ID}}, C)$: To decrypt $C = \langle U, V\rangle \in \mathcal{C}$ with the private key d_{ID}, compute the plaintext

$$M = H_2(\hat{e}(d_{\mathrm{ID}}, U) \oplus V$$

Anon-IND-ID-CPA Security [11, 12, 13, 14] of BF01 consists of semantic security of plaintext and anonymity of ID both under adaptive-ID (ID) and chosen plaintext attacks (CPA).

Definition 2 (Anon-IND-ID-CPA Security of BF01 in RO model) *BF01 is Anon-IND-ID-CPA secure in RO model, if any PPT adversary A, associated to the following experiment* $EXP^{d}_{BF01,A}(1^k)$ *for* $d \in \{0,1\}$*, has a negligible advantage* $Adv^{Anon\text{-}IND\text{-}ID\text{-}CPA}_{BF01,A}$*, where*

$$Adv^{Anon\text{-}IND\text{-}ID\text{-}CPA}_{BF01,A} = Pr[EXP^{1}_{BF01,A}(1^k) = 1] - Pr[EXP^{0}_{BF01,A}(1^k) = 1]$$

$\mathrm{EXP}^{d}_{\mathrm{BF01},A}(1^k)$

$IDSet \leftarrow \emptyset$; $H_1 - List \leftarrow \emptyset \subseteq \{0,1\}^* \times \mathbb{G}$; $H_2 - List \leftarrow \emptyset \subseteq \mathbb{G}_t \times \{0,1\}^n$;

$\langle BF01.Pub, s\rangle \overset{\$}{\leftarrow} Setup(1^k)$, where $BF01.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}\rangle$;

$\{ID^*, M^*, \mathrm{state}\} \overset{\$}{\leftarrow} A^{H_1Q(\cdot),H_2Q(\cdot),ExtractQ(\cdot)}(\mathrm{find}, BF01.Pub)$;

If $d = 0$, set the challenging ciphertext $C^* \overset{\$}{\leftarrow} Encrypt(BF01.Pub, ID^*, M^*)$;

If $d = 1$, set the challenging ciphertext $C^* \overset{\$}{\leftarrow} \mathcal{C}$;

$d' \leftarrow A^{H_1Q(\cdot),H_2Q(\cdot),ExtractQ(\cdot)}(\mathrm{guess}, C^*, \mathrm{state})$;

If $ID^* \notin IDSet$, return d'; Otherwise return 0;

Oracle $H_1Q(\mathrm{ID})$

If $\langle \mathrm{ID}, * \rangle \notin H_1 - List$, pick $P \overset{\$}{\leftarrow} \mathbb{G}$ and add $\langle \mathrm{ID}, P\rangle$ into $H_1 - List$; Otherwise, select $\langle \mathrm{ID}, P\rangle \in H_1 - List$; Return P;

Oracle $H_2Q(Z \in \mathbb{G}_t)$

If $\langle Z, * \rangle \notin H_2 - List$, pick $Str \overset{\$}{\leftarrow} \{0,1\}^n$ and add $\langle Z, Str\rangle$ into $H_1 - List$; Otherwise, select $\langle Z, Str\rangle \in H_2 - List$; Return Str;

Oracle $ExtractQ(\mathrm{ID})$

$IDSet = IDSet \bigcup \{\mathrm{ID}\}$; Return $d_{\mathrm{ID}} \leftarrow Extract(\mathrm{ID}, s)$;

Referring to semantic security proof of plaintext of BF01 [7], it is easy to prove the following theorem [11].

Theorem 1 *Assume that CBDH assumption holds in* $BGen(1^k)$*, then BF01 is Anon-IND-ID-CPA secure in RO model.*

3 Public-Key Encryption with Cluster-based Keyword Search

PCS consists of the following five algorithms: 1. $PCS.Init$ generates the public and private parameters of any security parameter; 2. $PCS.RootGen$ one-way generates a root from a particular secret; 3. $PCS.Trapdoor$ randomized generates a keyword search trapdoor for a keyword; 4. PCS generates a cluster-based keyword searchable ciphertext of a keyword; 5. $PCS.Test$ verifies that whether a cluster-based keyword searchable ciphertext contains a queried keyword. Their details are as follows:

- $PCS.Init(1^k)$: Given a security parameter 1^k, compute $\langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}\rangle \overset{\$}{\leftarrow} BGen(1^k)$; Pick $s \overset{\$}{\leftarrow} \mathbb{Z}^*_q$ and set $P_{pub} = g^s$; Choose two cryptographic hash functions $H_1 : \{0,1\}^* \rightarrow \mathbb{G}$ and

$H_2 : \mathbb{G}_t \to \{0,1\}^n$; Pick square matrix $\mathbf{OM} = \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix} \xleftarrow{\$} \mathbb{Z}_q^{m \times m}$ and compute $\mathbf{GM} = \begin{pmatrix} g^{r_{1,1}} & \cdots & g^{r_{1,m}} \\ \vdots & \ddots & \vdots \\ g^{r_{m,1}} & \cdots & g^{r_{m,m}} \end{pmatrix} \in \mathbb{G}^{m \times m}$. The keyword space is $\mathcal{W} = \{0,1\}^m - \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$. The cluster-based keyword searchable ciphertext space is $\mathcal{CL} = \{0,1\}^n$. The public parameter of PCS is $PCS.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, H_1, H_2, \mathbf{GM} \rangle$. The private parameter of PCS is $PCS.Pri = \langle s, \mathbf{OM}^{-1} \rangle$, which is the inverse square matrix of $\mathbf{OM}$.

- $PCS.RootGen(PCS.Pub, \mathbf{A})$: For a given private parameter $\mathbf{A} = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \xleftarrow{\$} \mathbb{Z}_q^{*\ m}$, compute PCS's root $\mathbf{Root} = \mathbf{A}^T \cdot \mathbf{GM} \in \mathbb{G}^{1 \times m}$ as

$$\begin{aligned} \mathbf{A}^T \cdot \mathbf{GM} &= \begin{pmatrix} a_1 & \cdots & a_m \end{pmatrix} \cdot \begin{pmatrix} g^{r_{1,1}} & \cdots & g^{r_{1,m}} \\ \vdots & \ddots & \vdots \\ g^{r_{m,1}} & \cdots & g^{r_{m,m}} \end{pmatrix} \\ &= \begin{pmatrix} \prod_{i \in [1,m]} g^{r_{i,1} \cdot a_i} & \cdots & \prod_{i \in [1,m]} g^{r_{i,m} \cdot a_i} \end{pmatrix} \\ &= \begin{pmatrix} g^{\sum_{i \in [1,m]} r_{i,1} \cdot a_i} & \cdots & g^{\sum_{i \in [1,m]} r_{i,m} \cdot a_i} \end{pmatrix} \end{aligned} \tag{1}$$

- $PCS.Trapdoor(W, PCS.Pri)$: For a given keyword $W = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix} \in \mathcal{W}$, pick $h \xleftarrow{\$} \mathbb{Z}_q^*$ and compute the keyword search trapdoor $t_W = \langle \mathbf{t}_W^1, t_W^2 \rangle$ as

$$\mathbf{t}_W^1 = \mathbf{OM}^{-1} \cdot W \cdot h \in \mathbb{Z}_q^m,\ t_W^2 = H_1(W)^{s \cdot h^{-1}}$$

- $PCS(PCS.Pub, \mathbf{A}, W)$: Given a keyword $W = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix} \in \mathcal{W}$ and a private parameter $\mathbf{A} = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \in \mathbb{Z}_q^{*\ m}$, compute the cluster-based keyword searchable ciphertext $CL_{\mathbf{A},W}$ as

$$CL_{\mathbf{A},W} = H_2(\hat{e}(P_{pub}, H_1(W))^{\sum_{i \in [1,m]} (w_i \cdot a_i)}) \in \mathcal{CL}$$

- $PCS.Test(\mathbf{Root}, CL_{\mathbf{A},W'}, t_W)$: Given a PCS's root $\mathbf{Root}$, a cluster-based keyword searchable ciphertext $CL_{\mathbf{A},W'}$ relevant to $\mathbf{Root}$ (namely, we have that $\mathbf{Root} = PCS.RootGen(PCS.Pub, \mathbf{A})$), and a keyword search trapdoor $t_W = \langle \mathbf{t}_W^1, t_W^2 \rangle$, if

$$CL_{\mathbf{A},W'} = H_2(\hat{e}(\mathbf{Root} \cdot \mathbf{t}_W^1, t_W^2))$$

then return 1, otherwise return 0.

Correctness of $PCS.Test(\mathbf{Root}, CL_{\mathbf{A},W'}, t_W)$. According to Equation 1, we have

$$\begin{aligned}\mathbf{Root} = \mathbf{A}^T \cdot \mathbf{GM} &= \begin{pmatrix} a_1 & \cdots & a_m \end{pmatrix} \cdot \begin{pmatrix} g^{r_{1,1}} & \cdots & g^{r_{1,m}} \\ \vdots & \ddots & \vdots \\ g^{r_{m,1}} & \cdots & g^{r_{m,m}} \end{pmatrix} \\ &= \begin{pmatrix} a_1 & \cdots & a_m \end{pmatrix} \cdot \begin{pmatrix} g & & 1 \\ & \ddots & \\ 1 & & g \end{pmatrix} \cdot \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix} \\ &= \begin{pmatrix} g^{a_1} & \cdots & g^{a_m} \end{pmatrix} \cdot \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix}\end{aligned} \tag{2}$$

Let $t_W = \langle \mathbf{t}^1_W, t^2_W \rangle$, where $\mathbf{t}^1_W = \mathbf{OM}^{-1} \cdot W \cdot h \in \mathbb{Z}_q^m$ and $t^2_W = H_1(W)^{s \cdot h^{-1}}$. According to Equation 2, we have

$$\begin{aligned}\mathbf{Root} \cdot \mathbf{t}^1_W &= \begin{pmatrix} g^{a_1} & \cdots & g^{a_m} \end{pmatrix} \cdot \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix} \cdot \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix}^{-1} \cdot \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix} \cdot h \\ &= \begin{pmatrix} g^{a_1} & \cdots & g^{a_m} \end{pmatrix} \cdot \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix} \cdot \begin{pmatrix} w_1 \cdot h \\ \vdots \\ w_m \cdot h \end{pmatrix} = g^{h \cdot \sum_{i \in [1,m]} (w_i \cdot a_i)}\end{aligned}$$

Furthermore, we have

$$H_2(\hat{e}(\mathbf{Root} \cdot \mathbf{t}^1_W, t^2_W)) = H_2(\hat{e}(g^{h \cdot \sum_{i \in [1,m]} (w_i \cdot a_i)}, H_1(W)^{s \cdot h^{-1}})) = H_2(\hat{e}(P_{pub}, H_1(W))^{\sum_{i \in [1,m]} (w_i \cdot a_i)})$$

Therefore, for $CL_{\mathbf{A},W'} \leftarrow PCS(PCS.Pub, \mathbf{A}, W')$, $W' = W$ holds if and only if $CL_{\mathbf{A},W'} = H_2(\hat{e}(\mathbf{Root} \cdot \mathbf{t}^1_W, t^2_W))$.

3.1 SS-CKA Security of PCS

The essence of PCS's SS-CKA security is the same with PEKS's S-CKA security. Their main differences are some adaptive revision by the differences between PCS and PEKS. For example, the SS-CKA security of PCS additionally allows an adversary to query a root oracle *RootQ*; The challenging keywords additionally attach with their roots.

Definition 3 (SS-CKA Security of PCS in RO model) *PCS is SS-CKA secure in RO model, if any PPT adversary A, associated to the following experiment $EXP^d_{PCS,A}(1^k)$ for $d \in \{0,1\}$, has a negligible advantage $Adv^{SS\text{-}CKA}_{PCS,A}$, where*

$$Adv^{SS\text{-}CKA}_{PCS,A} = Pr[EXP^1_{PCS,A}(1^k) = 1] - Pr[EXP^0_{PCS,A}(1^k) = 1]$$

$\mathrm{EXP}^d_{\mathrm{PCS},A}(1^k)$

Set $WSet \leftarrow \emptyset$, $H_1 - List \leftarrow \emptyset \subseteq \{0,1\}^* \times \mathbb{G}$, $H_2 - List \leftarrow \emptyset \subseteq \mathbb{G}_t \times \{0,1\}^n$ and $Root - List \leftarrow \emptyset \subseteq \mathbb{G}^m \times {\mathbb{Z}_q^*}^m$;

$\langle PCS.Pub, PCS.Pri\rangle \xleftarrow{\$} Init(1^k)$, where $PCS.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, \mathbf{GM}\rangle$ and $PCS.Pri = \langle s, \mathbf{OM}^{-1}\rangle$;

$\{\langle W_0, \mathbf{Root}_0\rangle, \langle W_1, \mathbf{Root}_1\rangle, \text{state}\} \xleftarrow{\$} A^{H_1Q(\cdot),H_2Q(\cdot),RootQ(\cdot),TrapQ(\cdot)}(\text{find}, PCS.Pub)$;

Select $\langle \mathbf{Root}_d, \mathbf{A}_d\rangle \in Root - List$;

Set the challenging cluster-based keyword searchable ciphertext $CL^* \leftarrow PCS(PCS.Pub, \mathbf{A}_d, W_d)$;

$d' \leftarrow A^{H_1Q(\cdot),H_2Q(\cdot),RootQ(\cdot),TrapQ(\cdot)}(\text{guess}, CL^*, \text{state})$;

If $\{W_0, W_1\} \nsubseteq WSet$, return d'; Otherwise return 0;

Oracle $H_1Q(W)$

If $\langle W, * \rangle \notin H_1 - List$, pick $P \xleftarrow{\$} \mathbb{G}$ and add $\langle W, P\rangle$ into $H_1 - List$; Otherwise, select $\langle W, P\rangle \in H_1 - List$; Return P;

Oracle $H_2Q(Z \in \mathbb{G}_t)$

If $\langle Z, * \rangle \notin H_2 - List$, pick $Str \xleftarrow{\$} \{0,1\}^n$ and add $\langle Z, Str\rangle$ into $H_1 - List$; Otherwise, select $\langle Z, Str\rangle \in H_2 - List$; Return Str;

Oracle $RootQ(NULL)$

Pick $\mathbf{A} \xleftarrow{\$} {\mathbb{Z}_q^*}^m$; $\mathbf{Root} \leftarrow PCS.RootGen(PCS.Pub, \mathbf{A})$; Add $\langle \mathbf{Root}, \mathbf{A}\rangle$ into $Root - List$; Return $\mathbf{Root}$;

Oracle $TrapQ(W)$

$WSet = WSet \bigcup \{W\}$; $t_W \xleftarrow{\$} PCS.Trapdoor(W, PCS.Pri)$; Return t_W;

Theorem 2 *Assume that CBDH assumption holds in $BGen(1^k)$, Our PCS is SS-CKA secure in RO model.*

4 Public-Key Encryption with Cluster-Chain-based Keyword Search

PCCS=$\langle PCCS.Init, PCCS.RootGen, PCCS.Trapdoor, PCCS, PCCS.Test\rangle$ is a combination of PCS and BF01. It uses BF01 to achieve public-key encryption with chain-based keyword search. Therefore, $PCCS.Init$ and $PCCS.RootGen$ are the same with $PCS.Init$ and $PCS.RootGen$. $PCCS.Trapdoor$ extends $PCS.Trapdoor$ to additionally output a chain-based keyword search trapdoor. $PCCS$ extends PCS to generate a cluster-chain-based keyword searchable ciphertext. $PCCS.Test$ extends $PCS.Test$ to verify that whether a cluster-chain-based keyword searchable ciphertext contains a queired keyword. The details are as follows:

- $PCCS.Init(1^k)$: Take as input a security parameter 1^k, compute $\langle PCS.Pub, PCS.Pri\rangle \xleftarrow{\$} PCS.Init(1^k)$, where $PCS.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, H_1, H_2, \mathbf{GM}\rangle$ and $PCS.Pri = \langle s, \mathbf{OM}^{-1}\rangle$.

The keyword space is $\mathcal{W} = \{0,1\}^m - \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$. Define a space $\mathcal{CH} = \{0,1\}^n \cup \{NULL\}$. The cluster-chain-based keyword searchable ciphertext space is $\mathcal{CC} = \{0,1\}^n \times \mathbb{G} \times \{0,1\}^n$. The public parameter of PCCS is $PCCS.Pub = PCS.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, H_1, H_2, \mathbf{GM} \rangle$. The private parameter of PCCS is $PCCS.Pri = PCS.Pri = \langle s, \mathbf{OM}^{-1} \rangle$.

- $PCCS.RootGen(PCCS.Pub, \mathbf{A})$: For a given private parameter $\mathbf{A} = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \xleftarrow{\$} {\mathbb{Z}_q^*}^m$, compute $\mathbf{Root} \leftarrow PCS.RootGen(PCCS.Pub, \mathbf{A})$.

- $PCCS.Trapdoor(W, PCCS.Pri)$: For a given keyword $W = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix} \in \mathcal{W}$, compute $\langle \mathbf{t}_W^1, t_W^2 \rangle \xleftarrow{\$} PCS.Trapdoor(W, PCCS.Pri)$; Compute $t_W^3 = H_1(W)^s$. The keyword search trapdoor of PCCS is $t_W = \langle \mathbf{t}_W^1, t_W^2, t_W^3 \rangle$.

- $PCCS(PCCS.Pub, \mathbf{A}, W, CH_{\mathbf{A},W})$: Take as input a private parameter $\mathbf{A} = \begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \in {\mathbb{Z}_q^*}^m$, a keyword $W = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix} \in \mathcal{W}$ and a global variable $CH_{\mathbf{A},W} \in \mathcal{CH}$, which is owned by the owner of $\mathbf{A}$ and initialized to be $NULL$; Pick $CH'_{\mathbf{A},W} \xleftarrow{\$} \{0,1\}^n$; This algorithm produces a cluster-chain-based keyword searchable ciphertexts $CC_{\mathbf{A},W} \in \mathcal{CC}$ by following steps:

  1. If $CH_{\mathbf{A},W} = NULL$ (it means that it is the first time to produce a cluster-chain-based keyword searchable ciphertext of W), set

     $$CC_{\mathbf{A},W} \xleftarrow{\$} CL_{\mathbf{A},W} || Encrypt(PCCS.Pub, W, CH'_{\mathbf{A},W})$$
     $$CH_{\mathbf{A},W} \leftarrow CH'_{\mathbf{A},W}$$

     where $CL_{\mathbf{A},W} = PCS(PCCS.Pub, \mathbf{A}, W)$;
  2. Otherwise, set

     $$CC_{\mathbf{A},W} \xleftarrow{\$} CH_{\mathbf{A},W} || Encrypt(PCCS.Pub, W, CH'_{\mathbf{A},W}(W))$$
     $$CH_{\mathbf{A},W} = CH'_{\mathbf{A},W}$$

- $PCCS.Test(\mathbf{Root}, CC_{\mathbf{A},W'}, t_W, V_{t_W,\mathbf{Root}})$: Take as input a PCCS's root $\mathbf{Root}$, a cluster-chain-based keyword searchable ciphertext $CC_{\mathbf{A},W'}$, a keyword search trapdoor $t_W = \langle \mathbf{t}_W^1, t_W^2, t_W^3 \rangle$, and a global viable $V_{t_W,\mathbf{Root}} \in \mathcal{CH}$ ($V_{t_W,\mathbf{Root}}$ is owned by that who does keyword search and initialized to be $NULL$); Parse $CC_{\mathbf{A},W'}$ as $\langle U \in \{0,1\}^n, C \in \mathbb{G} \times \{0,1\}^n \rangle$; This algorithm does the following two steps:

1. If $V_{t_W,\mathbf{Root}} = NULL$, then
   (a) if $PCS.Test(\mathbf{Root}, U, \langle \mathbf{t}_W^1, t_W^2 \rangle) = 1$, then set $V_{t_W,\mathbf{Root}} \leftarrow Decrypt(t_W^3, C)$ and return 1;
   (b) Otherwise return 0;
2. If $V_{t_W,\mathbf{Root}} = U$, then set $V_{t_W,\mathbf{Root}} \leftarrow Decrypt(t_W^3, C)$ and return 1; Otherwise return 0;

Correctness of $PCCS.Test(\mathbf{Root}, CC_{\mathbf{A},W'}, t_W, V_{t_W,\mathbf{Root}})$. We respectively demonstrate the correctness under the following two cases:

- Case 1: $V_{t_W,\mathbf{Root}} = NULL$. In this case, it is clear that the correctness of $PCCS.Test(\mathbf{Root}, CC_{\mathbf{A},W'}, t_W, V_{t_W,\mathbf{Root}})$ relies on the correctness of $PCS.Test(\mathbf{Root}, U, \langle \mathbf{t}_W^1, t_W^2 \rangle)$. Therefore, if $PCS.Test(\mathbf{Root}, U, \langle \mathbf{t}_W^1, t_W^2 \rangle) = 1$, then $W = W'$.
- Case 2: $V_{t_W,\mathbf{Root}} = U$. In this case, we show that the probability of $W \neq W'$ is negligible. Without loss of generality, let $V_{t_W,\mathbf{Root}} = CH'_{\mathbf{A},W}$. According to the algorithm $PCCS$, $CH'_{\mathbf{A},W}$ was randomly chosen in producing a cluster-chain-based keyword searchable ciphertext of W, and served as the first part of only one cluster-chain-based keyword searchable ciphertext of W. Therefore, if $W' \neq W$, $V_{t_W,\mathbf{Root}}$ is independent with U. In other words, the probability of $V_{t_W,\mathbf{Root}} = U$ is negligible, which contradict the case that $V_{t_W,\mathbf{Root}} = U$ holds. Under the case that $V_{t_W,\mathbf{Root}} = U$, the probability of $W \neq W'$ consequently is negligible.

4.1 SS-CKA Security of PCCS

The details of PCCS's SS-CKA security is the same with the SS-CKA of PCS, except that PCCS additionally provides an adversary the $PCCSQ$ oracle to query the cluster-chain-based keyword searchable ciphertext of any keyword.

Definition 4 (SS-CKA Security of PCCS in RO model) *$PCCS$ is SS-CKA secure in RO model, if any PPT adversary A, associated to the following experiment $EXP_{PCCS,A}^{d}(1^k)$ for $d \in \{0,1\}$, has a negligible advantage $Adv_{PCCS,A}^{SS\text{-}CKA}$, where*

$$Adv_{PCCS,A}^{SS\text{-}CKA} = Pr[EXP_{PCCS,A}^{1}(1^k) = 1] - Pr[EXP_{PCCS,A}^{0}(1^k) = 1]$$

$\mathrm{EXP}_{\mathrm{PCCS},A}^{d}(1^k)$

Set $WSet \leftarrow \emptyset$, $H_1 - List \leftarrow \emptyset \subseteq \{0,1\}^* \times \mathbb{G}$, $H_2 - List \leftarrow \emptyset \subseteq \mathbb{G}_t \times \{0,1\}^n$, $Root - List \leftarrow \emptyset \subseteq \mathbb{G}^m \times {\mathbb{Z}_q^*}^m$ and $CH - List \leftarrow \emptyset \subseteq {\mathbb{Z}_q^*}^m \times \mathcal{W} \times \{0,1\}^n \cup NULL$;

$\langle PCCS.Pub, PCCS.Pri \rangle \xleftarrow{\$} Init(1^k)$, where $PCCS.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, \mathbf{GM} \rangle$ and $PCCS.Pri = \langle s, \mathbf{OM}^{-1} \rangle$;

$\{\langle W_0, \mathbf{Root}_0 \rangle, \langle W_1, \mathbf{Root}_1 \rangle,\ \mathrm{state}\} \xleftarrow{\$} A^{H_1Q(\cdot),H_2Q(\cdot),RootQ(\cdot),TrapQ(\cdot),PCCSQ(\cdot)}(\mathrm{find}, PCCS.Pub)$;

Select $\langle \mathbf{Root}_d, \mathbf{A}_d \rangle \in Root - List$;

If $\langle \mathbf{A}_d, W_d, * \rangle \notin CH - List$, then set $CH_{\mathbf{A}_d,W_d} = NULL$;

Set the challenging cluster-chain-based keyword searchable ciphertext $CC^* \xleftarrow{\$} PCCS(PCCS.Pub, \mathbf{A}_d, W_d, CH_{\mathbf{A}_d,W_d})$;

Add $\langle \mathbf{A}_d, W_d, CH_{\mathbf{A}_d,W_d} \rangle$ into $CH-List$;
$d' \leftarrow A^{H_1Q(\cdot),H_2Q(\cdot),RootQ(\cdot),TrapQ(\cdot),PCCSQ(\cdot)}(\text{guess}, CC^*, \text{state})$;
If $\{W_0, W_1\} \nsubseteq WSet$, return d'; Otherwise return 0;

Oracle $H_1Q(W)$

If $\langle W, * \rangle \notin H_1-List$, pick $P \xleftarrow{\$} \mathbb{G}$ and add $\langle W, P \rangle$ into H_1-List; Otherwise, select $\langle W, P \rangle \in H_1-List$; Return P;

Oracle $H_2Q(Z \in \mathbb{G}_t)$

If $\langle Z, * \rangle \notin H_2-List$, pick $Str \xleftarrow{\$} \{0,1\}^n$ and add $\langle Z, Str \rangle$ into H_1-List; Otherwise, select $\langle Z, Str \rangle \in H_2-List$; Return Str;

Oracle $RootQ(NULL)$

Pick $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{*\,m}$; $\mathbf{Root} \leftarrow PCCS.RootGen(PCCS.Pub, \mathbf{A})$; Add $\langle \mathbf{Root}, \mathbf{A} \rangle$ into $Root-List$; Return $\mathbf{Root}$;

Oracle $TrapQ(W)$

$WSet = WSet \bigcup \{W\}$; $t_W \xleftarrow{\$} PCCS.Trapdoor(W, PCCS.Pri)$; Return t_W;

Oracle $PCCSQ(\mathbf{Root}, W)$

Select $\langle \mathbf{Root}, \mathbf{A} \rangle \in Root-List$; If $\langle \mathbf{A}, W, * \rangle \notin CH-List$, set $CH_{\mathbf{A},W} = NULL$;
$CC_{\mathbf{A},W} \xleftarrow{\$} PCCS(PCCS.Pub, \mathbf{A}, W, CH_{\mathbf{A},W})$;
Add $\langle \mathbf{A}, W, CH_{\mathbf{A},W} \rangle$ into $CH-List$; Return $CC_{\mathbf{A},W}$;

Theorem 3 *Assume that BF01 and our PCS respectively are Anon-IND-ID-CPA secure and SS-CKA secure both in RO model, Our PCCS is SS-CKA secure in RO model.*

According to Theorem 1,2 and 3, it is easy to prove the following theorem.

Theorem 4 *Assume CBDH assumption holds in $BGen(1^k)$, our PCCS is SS-CKA secure in RO model*

5 Performance Evaluation of PCCS

In this section, we show that our PCCS sharply enhance the keyword search performance, when the size of public parameter and the space complexity of cluster-chain-based keyword searchable ciphertext are slightly increased. Referring to Figure 1, we respectively suppose two fair application environments for PCCS and PEKS. Suppose that PCCS and PEKS have the same keyword space $\mathcal{W}$, where $|\mathcal{W}| = 2^m$; there are N senders in both environments; for all $i \in [1, N]$, sender S_i has generated T_{S_i,W_j} cluster-chain-based keyword searchable ciphertexts of keyword $W_j \in \mathcal{W}$ in PCCS; fairly, S_i in PEKS has generated T_{S_i,W_j} keyword searchable ciphertexts of keyword $W_j \in \mathcal{W}$. To more clearly show the advantages of PCCS, we further suppose that for all $i \in [1, N]$ and $W_j \in \mathcal{W}$, T_{S_i,W_j} are equal to T in both PCCS and PEKS.

5.1 High Keyword Search Performance

To respond a query on keyword $W_j \in \mathcal{W}$, the keyword search complexity of PCCS clearly relies on the times to compute bilinear map $\hat{e}$ and equals $\sum_{i\in[1,N]} T_{S_i,W_j} = N \cdot T$. For the same query, the keyword search complexity of BF01-based PEKS [1], which so far is the most efficient realization of PEKS, is equal to $\sum_{W_j\in\mathcal{W}} \sum_{i\in[1,N]} T_{S_i,W_j} = 2^m N \cdot T$. Obviously, our PCCS is much more efficient than PEKS.

5.2 Slight Increase in the Size of Public Parameter

In PCCS, the increase in the size of public parameter than BF01-based PEKS relies on the size of the parameter **GM**. Therefore, the increase is equal to $O((\log|\mathcal{W}|)^2) = O(m^2)$. Because m is the binary length of $|\mathcal{W}|$, we think that the increase would be slight.

5.3 Slight Increase in the Space Compelxity

In PEKS, the space complexity of keyword searchable ciphertexts is $O(2^m N \cdot T)$. In PCCS, the increase in the space complexity of cluster-chain-based keyword searchable ciphertexts than BF01-based PEKS relies on the size of PCCS's root. Therefore, the increase is equal to $O(N \cdot m)$. Relative to $O(2^m N \cdot T)$, the increase $O(N \cdot m)$ obviously is slight.

6 Summary

We novelly proposed public-key encryption with cluster-chain-based keyword search (PCCS), which performs much higher efficiency than PEKS, but also has the same security with PEKS. In random oracle (RO) model, we proved PCCS's semantic security of keywords under chosen keyword attacks (SS-CKA) from computational bilinear Diffie-Hellman (CBDH) assumption. In addition, PCCS just slightly increases the size of public parameter and space complexity of cluster-chain-based keyword searchable ciphertexts.

References

[1] D. Boneh, G. D. Crescenzo, R. Ostrovsky and G. Persiano. Public Key Encyrption with Keyword Search. In Advances in Cryptology-EUROCRYPT 2004, C. Cachin and J. Camenisch (ed.), Springer-Verlag, LNCS, 3027, 2004, 506–522

[2] D. X. Song, D. Wagner and A. Perrig. Practical techniques for searches on encrypted data. In Proceedings of the 2000 IEEE Symposium on Security and Privacy, 2000, 44–55

[3] E.-J. Goh. Secure Indexes. Technique Report 2003/216, IACR ePrint Cryptography Archive, 2003. See http://eprint.iacr.org/2003/216

[4] R. Agrawal, J. Kiernan, R. Srikant and Y. Xu. Order Preserving Encryption for Numeric Data. In Proceedings of the 2004 ACM SIGMOD international conference on Management of data, ACM, 2004, 563–574

[5] R. Curtmola, J. Garay, S. Kamara and R. Ostrovsky. Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. In Proceedings of the 13th ACM conference on Computer and communications security (ed.), ACM, 2006, 79–88

[6] 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 Advances in Cryptology-CRYPTO 2005, V. Shoup (ed.), Santa Barbara, California, United States, Springer-Verlag, LNCS, 3621, 2005, 205-222.

[7] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. In Advances in Cryptology-CRYPTO 2001, J. Kilian (ed.), Santa Barbara, California, United States, Springer-Verlag, LNCS, 2139, 2001, 213-239.

[8] D. Boneh and X. Boyen. Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In Advances in Cryptology-EUROCRYPT 2004, C. Cachin and J. Camenisch (ed.), Switzerland, Springer-Verlag, LNCS, 3027, 2004, 223-238.

[9] D. Boneh and X. Boyen. Secure Identity Based Encryption Without Random Oracles. In Advances in Cryptology-CRYPTO 2004, M. Franklin (ed.), Santa Barbara, California, United States, Springer-Verlag, LNCS, 3152, 2004, 197-206.

[10] B. Waters. Efficient Identity-Based Encryption Without Random Oracles. In Advances in Cryptology-EUROCRYPT 2005, R. Cramer (ed.), Denmark, Springer-Verlag, LNCS, 3494, 2005, 114-127.

[11] X. Boyen and B. Waters. Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In Advances in Cryptology-CRYPTO 2006, C. Dwork (ed.), Santa Barbara, California, United States, Springer-Verlag, LNCS, 4117, 2006, 290-307.

[12] C. Gentry. Practical Identity-Based Encyrption Without Random Oracles. In Advances in Cryptology-EUROCRYPT 2006, S. Vaudenay (ed.), Russia, Springer-Verlag, LNCS, 4004, 2006, 445-464.

[13] G. Ateniese and P. Gasti. Universally Anonymous IBE Based on the Quadratic Residuosity Assumption. In CT-RSA 2009, M. Fischlin (ed.), Springer-Verlag, LNCS, 5473, 2009, 32-47.

[14] L. Ducas. Anonymity from Asymmetry: New Constructions for Anonymous HIBE. In CT-RSA 2010, J. Pieprzyk (ed.), San Francisco, CA, USA, Springer Berlin, LNCS, 5985, 2010, 148-164.

[15] A. J. Menezes, T. Okamoto and S. A. Vanstone. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. IEEE TRANSACTIONS ON INFORMATION THEORY, 1993, 39(5): 1639-1646.

[16] G. Frey, M. Muller and H.-G. Ruck. The Tate Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems. IEEE TRANSACTIONS ON INFORMATION THEORY, 1999, 45(5): 1717-1719.

[17] R. Curtmola, J. Garay, S. Kamara and R. Ostrovsky. Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. In Proceedings of the 13th ACM conference on Computer and communications security, ACM, 2006, 79-88.

[18] Y.-C. Chang and M. Mitzenmacher. Privacy Preserving Keyword Searches on Remote Encrypted Data. In Applied Cryptography and Network Security 2005, Springer-Verlag, LNCS, 3531, 2005, 391-421.

[19] S.M. Bellovin and W.R. Cheswick. Privacy-enhanced searches using encrypted Bloom filters. Technical Report 2004/022, IACR ePrint Cryptography Archive, 2004.

[20] R. Agrawal, J. Kiernan, R. Srikant and Y. Xu. Order Preserving Encryption for Numeric Data. In Proceedings of the 2004 ACM SIGMOD international conference on Management of data, ACM, 2004, 563-574.

[21] A. Boldyreva, N. Chenette, Y. Lee and A. ONeill. Order-Preserving Symmetric Encryption. In Advances in Cryptology-EUROCRYPT 2009, A. Joux (ed.), Springer-Verlag, LNCS, 5749, 2009, 224-241.

[22] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren and W. Lou. Fuzzy Keyword Search over Encrypted Data in Cloud Computing. In Proceedings IEEE INFOCOM 2010, 2010, 441-445.

[23] F. Bao, R. H. Deng, X. Ding and Y. Yang. Private query on encrypted data in multi-user settings. In ISPEC'08 Proceedings of the 4th international conference on Information security practice and experience, Springer-Verlag, LNCS, 4991, 2008, 71-85.

[24] B. Waters, D. Balfanz, G. Durfee and D. K. Smetters. Building an Encrypted and Searchable Audit Log. In The 11th Annual Network and Distributed System Security Symposium, 2004.

[25] J. Baek, R. Safavi-Naini and W. Susilo. Public Key Encryption with Keyword Search Revisited. In Computational Science and Its Applications-ICCSA 2008 O. Gervasi (ed.), Springer-Verlag, 5072, 2008, 1249-1259.

[26] D. J. Park, K. Kim and P. J. Lee. Public Key Encryption with Conjunctive Field Keyword Search. In WISA 2004, C. H. Lim and M. Yung (ed.), Spring-Verlag, LNCS, 3325, 2004, 73-86.

[27] P. Golle, J. Staddon and B. Waters. Secure conjunctive keyword search over encrypted data. In 2nd International Conference on Applied Cryptography and Network Security, Springer-Verlag, LNCS, 3089, 2004, 31-45.

[28] D. Boneh and B. Waters. conjunctive, subset, and range queries on ecrypted data. In proceedings of TCC'07 S. P. Vadhan (ed.), Springer-Verlag, LNCS, 4392, 2007, 535-554.

[29] Y. Zhang, N. Mamoulis, D. W. Cheung, S. M. Yiu and W. K. Wong. Anonymous Fuzzy Identity-based Encryption for Similarity Search. In ISAAC 2010, Springer-Verlag, LNCS, 6505, 2010, 61-72.

[30] Y. H. Hwang and P. J. Lee. Public Key Encryption with Conjunctive Keyword Search and Its Extension to a Multi-user System. In Pairing 2007, T. Takagi (ed.), Springer-Verlag, LNCS, 4575, 2007, 2-22.

[31] L. Ballard, S. Kamara and F. Monrose. Achieving Efficient Conjunctive Keyword Searches over Encrypted Data. In ICICS 2005, Springer-Verlag, LNCS, 3783, 2005, 414-426.

[32] E.K. Ryu and T. Takagi. Efficient Conjunctive Keyword-Searchable Encryption. In 21st International Conference on Advanced Information Networking and Applications Workshops, IEEE Computer Society, 2007, 409 - 414

[33] J. Bethencourt, T.-H. H. Chan, A. Perrig, E. Shi and D. Song. Anonymous multi-attribute encryption with range query and conditional decryption. CMU-CS-06-135, 2006.

[34] E. Shi, J. Bethencourt, T.-H. H. Chan, D. Song and A. Perrig. Multi-Dimensional Range Query over Encrypted Data. In IEEE Symposium on Security and Privacy, 2007, 350 - 364.

[35] J. Camenisch, M. Kohlweiss, A. Rial and C. Sheedy. Blind and Anonymous Identity-Based Encryption and Authorised Private Searches on Public Key Encrypted Data. In Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC '09, CA, Springer-Verlag, LNCS, 5443, 2009, 196-214.

[36] M. Bellare, A. Boldyreva and A. ONeill. Deterministic and Efficiently Searchable Encryption. In Advances in Cryptology - CRYPTO 2007, A. Menezes (ed.), Santa Barbara, California, United States, Springer-Verlag, LNCS, 4622, 2007, 535-552.

[37] M. BELLARE, M. FISCHLIN, A. ONEILL and T. RISTENPART. Deterministic Encryption: Definitional Equivalences and Constructions without Random Oracles. In Advances in Cryptology-CRYPTO08, D. Wagner (ed.), Springer-Verlag, LNCS, 5157, 2008, 360-378.

[38] A. Boldyreva, S. Fehr and A. ONeill. On Notions of Security for Deterministic Encryption, and Efficient Constructions without Random Oracles. In Advances in Cryptology-CRYPTO 2008, D. Wagner (ed.), Springer-Verlag, LNCS, 5157, 2008, 335-359.

[39] L. Fang, W. Susilo, C. Ge and J. Wang. A Secure Channel Free Public Key Encryption with Keyword Search Scheme without Random Oracle. In Cryptology and Network Security 2009, Springer-Verlag, LNCS, 5888, 2009, 248-258.

[40] D. Davis, F. Monrose and M. K. Reiter. Time-Scoped Searching of Encrypted Audit Logs. In ICICS 2004, Springer-Verlag, LNCS, 3269, 2004, 532-545.

[41] M. Hattori, T. Hirano, T. Ito, N. Matsuda, T. Mori, Y. Sakai and K. Ohta. Ciphertext-Policy Delegatable Hidden Vector Encryption and Its Application to Searchable Encryption in Multi-user Setting. In Cryptography and Coding 2011, Springer-Verlag, LNCS, 7089, 2011, 190-209.

[42] J. Baek, R. Safavi-Naini and W. Susilo. On the Integration of Public Key Data Encryption and Public Key Encryption with Keyword Search. In ISC 2006, Springer-Verlag, LNCS, 4176, 2006, 217-232.

[43] R. Zhang and H. Imai. Generic Combination of Public Key Encryption with Keyword Search and Public Key Encryption. In CRYPTOLOGY AND NETWORK SECURITY 2007, Springer-Verlag, LNCS, 4856, 2007, 159-174.

[44] J. Camenisch, M. Kohlweiss, A. Rial and C. Sheedy. Blind and Anonymous Identity-Based Encryption and Authorised Private Searches on Public Key Encrypted Data. In Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC '09, CA, Springer-Verlag, LNCS, 5443, 2009, 196-214.

A Proof of Theorem 2

Supposing A has the non-negligible advantage $\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$ to break PCS's SS-CKA security defined in Definition 3. We define an adversary B, who leverages A to solve CBDH problem defined in Subsection 2.2. Generate $\langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}\rangle \xleftarrow{\$} BGen(1^k)$. B takes as input $\langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, g^a, g^b, g^c\rangle$, where $\langle g^a, g^b, g^c\rangle \xleftarrow{\$} \mathbb{G}^3$. The more details of B are as follows.

Adversary $B(q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, g^a, g^b, g^c)$

Set $WSet \leftarrow \emptyset$, $H_1-List \leftarrow \emptyset \subseteq \mathcal{W}\times\mathbb{G}\times\mathbb{Z}_q^*\times\{0,1\}$, $H_2-List \leftarrow \emptyset \subseteq \mathbb{G}_t\times\{0,1\}^n$, and $Root-List \leftarrow \emptyset \subseteq \mathbb{G}^{1\times m}\times {\mathbb{Z}_q^*}^{m}$;

Pick $\mathbf{OM} = \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix} \xleftarrow{\$} {\mathbb{Z}_q^*}^{m\times m}$;

Compute $\mathbf{GM} = \begin{pmatrix} g^a & & 1 \\ & \ddots & \\ 1 & & g^a \end{pmatrix} \cdot \mathbf{OM} = \begin{pmatrix} g^{a\cdot r_{1,1}} & \cdots & g^{a\cdot r_{1,m}} \\ \vdots & \ddots & \vdots \\ g^{a\cdot r_{m,1}} & \cdots & g^{a\cdot r_{m,m}} \end{pmatrix}$;

Set $PCS.Pub \leftarrow \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, \mathbf{GM}\rangle$, where $P_{pub} \leftarrow g^b$;

$\{\langle W_0, \mathbf{Root}_0\rangle, \langle W_1, \mathbf{Root}_1\rangle, \text{state}\} \xleftarrow{\$} A^{H_1Q(\cdot),H_2Q(\cdot),RootQ(\cdot),TrapQ(\cdot)}(\text{find}, PCS.Pub)$;

If $\langle W_0, *, *\rangle \notin H_1-List$, query $H_1Q(W_0)$; If $\langle W_1, *, *\rangle \notin H_1-List$, query $H_1Q(W_1)$;

Select $\langle W_0, H_1(W_0), x_0, Coin_0\rangle \in H_1-List$, if $Coin_0 = 0$ then return $\perp$ and abort;

Select $\langle W_1, H_1(W_1), x_1, Coin_1\rangle \in H_1-List$, if $Coin_1 = 0$ then return $\perp$ and abort;

Pick $d \xleftarrow{\$} \{0,1\}$;

Set the challenging cluster-based keyword searchable ciphertext $CL^* \xleftarrow{\$} \{0,1\}^n$;

$d' \leftarrow A^{H_1Q(\cdot),H_2Q(\cdot),RootQ(\cdot),TrapQ(\cdot)}(\text{guess}, CL^*, \text{state})$;

Select $\langle \mathbf{Root}_0, \mathbf{A}_0\rangle \in Root-List$; let $W_0 = \begin{pmatrix} w_1^0 \\ \vdots \\ w_m^0 \end{pmatrix}$ and $\mathbf{A}_0 = \begin{pmatrix} a_1^0 \\ \vdots \\ a_m^0 \end{pmatrix}$;

Select $\langle \mathbf{Root}_1, \mathbf{A}_1\rangle \in Root-List$; let $W_1 = \begin{pmatrix} w_1^1 \\ \vdots \\ w_m^1 \end{pmatrix}$ and $\mathbf{A}_1 = \begin{pmatrix} a_1^1 \\ \vdots \\ a_m^1 \end{pmatrix}$;

If $\{W_0, W_1\} \nsubseteq WSet$, then randomly select $\langle Z, Str\rangle \in H_2-List$ and return $Z^{(x_d\cdot\sum_{i\in[1,m]}(w_i^d\cdot a_i^d))^{-1}}$;

Oracle $H_1Q(W)$

Pick $Coin \xleftarrow{\delta} \{0,1\}$ according to the probability $\Pr[Coin = 1] = \delta$, in which δ will be decided latter;

If $Coin = 1$, set $H_1(W) \leftarrow g^{c\cdot x}$; Otherwise set $H_1(W) \leftarrow g^x$, where $x \xleftarrow{\$} \mathbb{Z}_q^*$;

Add $\langle W, H_1(W), x, Coin\rangle$ into H_1-List;

return $H_1(W)$;

Oracle $H_2Q(Z)$

For a inputted $Z \in \mathbb{G}_t$, pick $Str \xleftarrow{\$} \{0,1\}^n$; Add $\langle Z, Str\rangle$ into H_2-List;

return Str;

Oracle $RootQ(NULL)$

Pick $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{*\ m}$; Set $\mathbf{Root} \leftarrow PCS.RootGen(PCS.Pub, \mathbf{A})$;
Add $\langle \mathbf{Root}, \mathbf{A} \rangle$ into $Root - List$;
return $\mathbf{Root}$;

Oracle $TrapQ(W)$

$WSet = WSet \bigcup \{W\}$;
Select $\langle W, H_1(W), x, Coin \rangle \in H_1 - List$; If $Coin = 1$ then return $\perp$ and abort;
Compute $\mathbf{t}_W^1 = \mathbf{OM}^{-1} \cdot W \cdot h$ and $t_W^2 = g^{b \cdot x \cdot h^{-1}}$, where $h \xleftarrow{\$} \mathbb{Z}_q^*$;
Return $t_W = \langle \mathbf{t}_W^1, t_W^2 \rangle$

B is simulating a real attack environment for A. Assume A makes at most the polynomial-q_W $TrapQ(\cdot)$ queries. We prove that B output $\hat{e}(g,g)^{abc}$ with probability at least $\frac{1}{(e \cdot q_W)^2} \cdot \mathrm{Adv}_{\mathrm{PCS},A}^{\text{SS-CKA}}$. Let $\mathcal{H}$ denote the event that A queries H_2Q for $\hat{e}(g,g)^{abc \cdot x_0 \cdot \sum_{i \in [1,m]} (w_i^0 \cdot a_i^0)}$ or $\hat{e}(g,g)^{abc \cdot x_1 \cdot \sum_{i \in [1,m]} (w_i^1 \cdot a_i^1)}$, in which $\hat{e}(g,g)^{abc \cdot x_d \cdot \sum_{i \in [1,m]} (w_i^d \cdot a_i^d)}$ is the correct input of H_2Q of CL^* in both the simulation and the real attack. The proof is analogous to the corresponding parts of BF01 [7].

Claim 1: If B does not abort, $\Pr[\mathcal{H}]$ in the simulation is equal to $\Pr[\mathcal{H}]$ in the real attack.

Proof of Claim 1. Let $\mathcal{H}_l$ denote the event that A queries H_2Q for $\hat{e}(g,g)^{abc \cdot x_0 \cdot \sum_{i \in [1,m]} (w_i^0 \cdot a_i^0)}$ or $\hat{e}(g,g)^{abc \cdot x_1 \cdot \sum_{i \in [1,m]} (w_i^1 \cdot a_i^1)}$ in the first l queries to the oracle H_2Q. We prove by induction on l that $\Pr[\mathcal{H}_l]$ in the real attack is equal to $\Pr[\mathcal{H}_l]$ in the simulation for all $l \geq 0$. Obviously, if B does not abort, $\Pr[\mathcal{H}_0] = 0$ in both the real attack and the simulation. Suppose that for some $l > 0$, $\Pr[\mathcal{H}_{l-1}]$ in the simulation is equal to $\Pr[\mathcal{H}_{l-1}]$ in the real attack. We prove that the same holds for $\mathcal{H}_l$.

$$\begin{aligned} \Pr[\mathcal{H}_l] &= \Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \cdot \Pr[\mathcal{H}_{l-1}] + \Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \cdot \Pr[\neg \mathcal{H}_{l-1}] \\ &= \Pr[\mathcal{H}_{l-1}] + \Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \cdot \Pr[\neg \mathcal{H}_{l-1}] \end{aligned} \tag{3}$$

We argue that $\Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \cdot \Pr[\neg \mathcal{H}_{l-1}]$ in the simulation is equal to $\Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \cdot \Pr[\neg \mathcal{H}_{l-1}]$ in the real attack. If B does not abort, all responses of oracles in the simulation have the same distribution with the real attack. $\neg \mathcal{H}_{l-1}$ means that CL^* in the simulation has the same distribution with the real attack. So if B does not abort and $\neg \mathcal{H}_{l-1}$ holds, then the view of A in the simulation is the same with the real attack during the first $l-1$ queries. Further, we have $\Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \cdot \Pr[\neg \mathcal{H}_{l-1}]$ in the simulation is equal to $\Pr[\mathcal{H}_l | \mathcal{H}_{l-1}] \cdot \Pr[\neg \mathcal{H}_{l-1}]$ in the real attack. By Equation 3 and induction on l, we obtain that if B does not abort, $\Pr[\mathcal{H}]$ in the simulation is equal to $\Pr[\mathcal{H}]$ in the real attack.

Claim 2: In the real attack, we have $\Pr[\mathcal{H}] = \frac{1}{2} \mathrm{Adv}_{\mathrm{PCS},A}^{\text{SS-CKA}}$.

Proof of Claim 2. In the real attack, if A never queries H_2Q for $\hat{e}(g,g)^{abc \cdot x_0 \cdot \sum_{i \in [1,m]} (w_i^0 \cdot a_i^0)}$ or $\hat{e}(g,g)^{abc \cdot x_1 \cdot \sum_{i \in [1,m]} (w_i^1 \cdot a_i^1)}$, CL^* is independent with d. Therefore, in the real attack we have

$\Pr[d = d'|\neg\mathcal{H}] = \frac{1}{2}$. By the assumption on A at the beginning of this section, we know that $\Pr[d = d'] = \frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A} + \frac{1}{2}$. In addition, the Equation 4 holds in the real attack.

$$\Pr[d = d'] = \Pr[d = d'|\neg\mathcal{H}] \cdot \Pr[\neg\mathcal{H}] + \Pr[d = d'|\mathcal{H}] \cdot \Pr[\mathcal{H}] = \frac{1}{2} + \Pr[\mathcal{H}] \tag{4}$$

It follows that we have $\Pr[\mathcal{H}] = \frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$.

The probability of the event $\overline{\mathbf{Abort}_B}$ that B does not abort. Because all responses of oracles $H_1Q(\cdot)$, $H_2Q(\cdot)$, $RootQ(\cdot)$ and $TrapQ(\cdot)$ are uniformly distributed in their spaces. For at most the polynomial-q_W $TrapQ(\cdot)$ queries, we have $\Pr[\overline{\mathrm{Abort}_B}] = \delta^2(1-\delta)^{q_W}$. When $\delta = \frac{2}{q_W+2}$, $\Pr[\overline{\mathrm{Abort}_B}] \approx (\frac{2}{e \cdot q_W})^2$ is the optimal value, in which e is the base of natural logarithm.

The probability that B outputs $\hat{e}(g,g)^{abc}$. When considering the probability of the event $\overline{\mathrm{Abort}_B}$, the probability $\Pr[\mathcal{H}]$ in the simulation is equal to $\frac{2}{(e \cdot q_W)^2} \cdot \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$. For the random selection $\langle Z, Str\rangle \in H_2 - List$ in the simulation, we have the probability at least $\frac{1}{(e \cdot q_W)^2} \cdot \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$ that $Z = \hat{e}(g,g)^{abc \cdot x_d \cdot \sum_{i\in[1,m]}(w_i^d \cdot a_i^d)}$. Therefore, B output $\hat{e}(g,g)^{abc}$ with the probability at least $\frac{1}{(e \cdot q_W)^2} \cdot \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$ by computing $Z^{(x_d \cdot \sum_{i\in[1,m]}(w_i^d \cdot a_i^d))^{-1}}$.

Our PCS is SS-CKA secure in RO model, when CBDH assumption holds in $BGen(1^k)$. According to above analysis, B has the advantage at least $\frac{1}{(e \cdot q_W)^2} \cdot \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$ to solve CBDH problem. When $\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$ is non-negligible, $\frac{1}{(e \cdot q_W)^2} \cdot \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$ also is non-negligible, which is contradictive with CBDH assumption holds in $BGen(1^k)$. Therefore, when CBDH assumption holds in $BGen(1^k)$, no adversary can break SS-CKA security of our PCS with non-negligible in RO model.

B Proof of Theorem 3

Supposing A has the non-negligible advantage $\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A}$ to break our PCCS's SS-CKA security defined in Definition 4. We define an adversary B, who employs A to break BF01's Anon-IND-ID-CPA security defined in Definition 2. Let $BF01.Pub = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}\rangle$ be the public parameter of BF01. Let $H_1Q(\cdot)$, $H_2Q(\cdot)$ and $ExtractQ(\cdot)$ be oracles provided by BF01 and defined in Definition 2. B takes as input $BF01.Pub$, issues queries to oracles $H_1Q(\cdot)$, $H_2Q(\cdot)$ and $ExtractQ(\cdot)$, and simulates a real attack environment of PCCS's SS-CKA security for A. The following description respectively are the details of B in 'find' and 'guess' phases of BF01's Anon-IND-ID-CPA security.

Adversary $B^{H_1Q(\cdot),H_2Q(\cdot),ExtractQ(\cdot)}(\text{find}, BF01.Pub)$

Set $WSet \leftarrow \emptyset$, $H_1 - List \leftarrow \emptyset \subseteq \{0,1\}^* \times \mathbb{G}$, $H_2 - List \leftarrow \emptyset \subseteq \mathbb{G}_t \times \{0,1\}^n$, $Root - List \leftarrow \emptyset \subseteq \mathbb{G}^m \times {\mathbb{Z}_q^*}^m$ and $CH - List \leftarrow \emptyset \subseteq {\mathbb{Z}_q^*}^m \times \mathcal{W} \times \{0,1\}^n \cup NULL$;

Pick $\mathbf{OM} = \begin{pmatrix} r_{1,1} & \cdots & r_{1,m} \\ \vdots & \ddots & \vdots \\ r_{m,1} & \cdots & r_{m,m} \end{pmatrix} \xleftarrow{\$} {\mathbb{Z}_q^*}^{m\times m}$; Compute $\mathbf{GM} = \begin{pmatrix} g^{r_{1,1}} & \cdots & g^{r_{1,m}} \\ \vdots & \ddots & \vdots \\ g^{r_{m,1}} & \cdots & g^{r_{m,m}} \end{pmatrix}$;

Set $PCCS.Pub = \langle BF01.Pub, \mathbf{GM}\rangle = \langle q, \mathbb{G}, \mathbb{G}_t, g, \hat{e}, P_{pub}, \mathbf{GM}\rangle$;

$\{\langle W_0, \mathbf{Root}_0\rangle, \langle W_1, \mathbf{Root}_1\rangle, \text{state}'\} \stackrel{\$}{\leftarrow} A^{H_1Q'(\cdot),H_2Q'(\cdot),RootQ(\cdot),TrapQ(\cdot),PCCSQ(\cdot)}(\text{find}, PCCS.Pub)$;

Pick $d' \stackrel{\$}{\leftarrow} \{0,1\}$ and $M \stackrel{\$}{\leftarrow} \{0,1\}^n$;

Return $\{W_{d'}, M, \text{state} = \{W_{d'}, \mathbf{Root}_{d'}, M\}\}$;

Adversary $B^{H_1Q(\cdot),H_2Q(\cdot),ExtractQ(\cdot)}(\text{guess}, C^*, \text{state} = \{W_{d'}, \mathbf{Root}_{d'}, M\})$

Select $\langle \mathbf{Root}_{d'}, \mathbf{A}_{d'}\rangle \in Root - List$;

If $\langle \mathbf{A}_{d'}, W_{d'}, *\rangle \notin CH - List$, then

Set $CH_{\mathbf{A}_{d'},W_{d'}} = NULL$;

Compute $CL_{\mathbf{A}_{d'},W_{d'}} = PCS(PCCS.Pub, \mathbf{A}_{d'}, W_{d'})$;

Set $CC^* \leftarrow CL_{\mathbf{A}_{d'},W_{d'}}||C^*$;

Otherwise

Select $\langle \mathbf{A}_{d'}, W_{d'}, CH_{\mathbf{A}_{d'},W_{d'}}\rangle \in CH - List$;

Set $CC^* \leftarrow CH_{\mathbf{A}_{d'},W_{d'}}||C^*$;

Set $CH_{\mathbf{A}_d,W_d} = M$;

Add $\langle \mathbf{A}_d, W_d, CH_{\mathbf{A}_d,W_d}\rangle$ into $CH - List$;

$d'' \leftarrow A^{H_1Q'(\cdot),H_2Q'(\cdot),RootQ(\cdot),TrapQ(\cdot),PCCSQ(\cdot)}(\text{guess}, CC^*, \text{state}')$;

If $\{W_0, W_1\} \nsubseteq WSet$, then

If $d'' = d'$, then return 0; Otherwise return 1;

Oracle $H_1Q'(W)$

If $\langle W, *\rangle \notin H_1 - List$, query $P \stackrel{\$}{\leftarrow} H_1Q(W)$ and add $\langle W, P\rangle$ into $H_1 - List$;

Otherwise, select $\langle W, P\rangle \in H_1 - List$;

Return P;

Oracle $H_2Q'(Z \in \mathbb{G}_t)$

If $\langle Z, *\rangle \notin H_2 - List$, query $Str \stackrel{\$}{\leftarrow} H_2Q(Z)$ and add $\langle Z, Str\rangle$ into $H_1 - List$;

Otherwise, select $\langle Z, Str\rangle \in H_2 - List$;

Return Str;

Oracle $RootQ(NULL)$

Pick $\mathbf{A} \stackrel{\$}{\leftarrow} \mathbb{Z}_q^{*\ m}$;

$\mathbf{Root} \leftarrow PCCS.RootGen(PCCS.Pub, \mathbf{A})$;

Add $\langle \mathbf{Root}, \mathbf{A}\rangle$ into $Root - List$;

Return $\mathbf{Root}$;

Oracle $TrapQ(W)$

$WSet = WSet \bigcup \{W\}$; Query $t_W^3 \leftarrow ExtractQ(W) \in \mathbb{G}$;

Pick $h \stackrel{\$}{\leftarrow} \mathbb{Z}_q^*$; Set $\mathbf{t}_W^1 = \mathbf{OM}^{-1} \cdot W \cdot h$ and $t_W^2 = (t_W^3)^{h^{-1}}$;

$t_W \leftarrow \langle \mathbf{t}_W^1, t_W^2, t_W^3\rangle$;

Return t_W;

Oracle $PCCSQ(\mathbf{Root}, W)$

Select $\langle \mathbf{Root}, \mathbf{A} \rangle \in Root - List$;
If $\langle \mathbf{A}, W, * \rangle \notin CH - List$, set $CH_{\mathbf{A},W} = NULL$;
$I \xleftarrow{\$} PCCS(PCCS.Pub, \mathbf{A}, W, CH_{\mathbf{A},W})$;
Add $\langle \mathbf{A}, W, CH_{\mathbf{A},W} \rangle$ into $CH - List$;
Return I;

B is simulating a real attack environment for A to break our PCCS's SS-CKA security. We prove that B has the advantage $\frac{1}{4}(\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A} + \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A})$ to break BF01's Anon-IND-ID-CPA security. Let $d \in \{0, 1\}$ be the random coin in the experiment of BF01's Anon-IND-ID-CPA security definition. Recall that when $b = 0$, the challenging ciphertext C^*, which was taken as input of B's 'guess' phase, is equal to $Encrypt(BF01.Pub, W_{d'}, M)$. Contrarily, when $d = 1$, C^* is random and independent with $\{W_0, W_1, M\}$.

Claim 1: If $d = 0$ namely that $C^* = Encrypt(BF01.Pub, W_{d'}, M)$, A correctly guesses d' (namely $d'' = d'$) with probability $\frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A} + \frac{1}{2}$ in the simulation of B.

Proof of Claim 1. Observe that all outputs of oracles provided to A by B have the same distribution with a real attack environment of PCCS's SS-CKA security. Further, when $C^* = Encrypt(BF01.Pub, W_{d'}, M)$, CC^* always be correct as a real one under both two cases that $CC^* = CL_{\mathbf{A}_{d'},W_{d'}}||C^*$ and $CC^* = CH_{\mathbf{A}_{d'},W_{d'}}||C^*$. Therefore, When A has the advantage $\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A}$ to break PCCS's SS-CKA security, A has the probability $\frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A} + \frac{1}{2}$ to correctly guess d' in the simulation of B.

Claim 2: If $d = 1$ namely that C^* is random, A correctly guesses d' (namely $d'' = d'$) with probability $\frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A} + \frac{1}{2}$ in the simulation of B.

Proof of Claim 2. Observe that all outputs of oracles provided to A by B are independent with d, so they also have the same distribution with a real attack environment of PCCS's SS-CKA security for both $d = 0$ and $d = 1$. According to BF01's Anon-IND-ID-CPA security definition, C^* is random and independent with $\{W_0, W_1, M\}$ when $d = 1$. We further analyze respectively under the two cases of CC^* that $CC^* = CL_{\mathbf{A}_{d'},W_{d'}}||C^*$ and $CC^* = CH_{\mathbf{A}_{d'},W_{d'}}||C^*$.

- Case 1: $CC^* = CL_{\mathbf{A}_{d'},W_{d'}}||C^*$. Because C^* is randomly chosen and independent with $W_{d'}$, the probability of $d'' = d'$ only relies on the advantage that A can correctly guess d' from $CL_{\mathbf{A}_{d'},W_{d'}}$. So when $CC^* = CL_{\mathbf{A}_{d'},W_{d'}}||C^*$, A correctly guesses d' with the probability $\frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A} + \frac{1}{2}$.
- Case 2: $CC^* = CH_{\mathbf{A}_{d'},W_{d'}}||C^*$. Because $CH_{\mathbf{A}_{d'},W_{d'}}$ and C^* are randomly chosen and independent with $W_{d'}$, the probability that A correctly guesses d' is $\frac{1}{2}$.

In summary, under the two cases that $CC^* = CL_{\mathbf{A}_{d'},W_{d'}}||C^*$ and $CC^* = CH_{\mathbf{A}_{d'},W_{d'}}||C^*$, the probability that A correctly guesses d' is equal to the maximum successful probability of that two cases. It is equal to $\frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A} + \frac{1}{2}$ in the simulation of B.

The advantage of B to break BF01's Anon-IND-ID-CPA security. According to Claim 1 and Claim 2, the probability that B correctly guesses d is equal to

$$(\frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A} + \frac{1}{2}) \cdot \Pr[d = 0] + (\frac{1}{2}\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A} + \frac{1}{2}) \cdot \Pr[d = 1] = \frac{1}{4}(\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A} + \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}) + \frac{1}{2}$$

In other words, B has the advantage $\frac{1}{4}(\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A} + \mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A})$ to break BF01's Anon-IND-ID-CPA security.

Our PCCS is SS-CKA secure in RO model, when BF01 and Our PCS respectively are Anon-IND-ID-CPA and SS-CKA secure. If any one of $\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCCS},A}$ and $\mathrm{Adv}^{\text{SS-CKA}}_{\text{PCS},A}$ is non-negligible, then the advantage of B to break BF01's Anon-IND-ID-CPA security also is non-negligible. But this conclusion contradicts that BF01 and Our PCS respectively are Anon-IND-ID-CPA and SS-CKA secure. Therefore, Our PCCS is SS-CKA secure in RO model, when BF01 and Our PCS respectively are Anon-IND-ID-CPA and SS-CKA secure.